Canarias, Ceuta y Melilla 335 ptas



REVISTA INDEPENDIENTE PARA USUARIOS DE ORDENADORES SINCLAIR

Nº 2 – 350 PTS

TE CONTAMOS TODO SOBRE LOS JUEGOS DE GUERRA

Qué son, para qué sirven y cómo se usan LAS VARIABLES DEL SISTEMA

Guía de las mejores utilidades para SPECTRUM

Así se hizo el "Camelot Warriors"



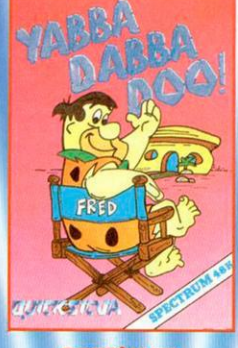
Conoce los lenguajes de tu Spectrum

HOBBY PRESS

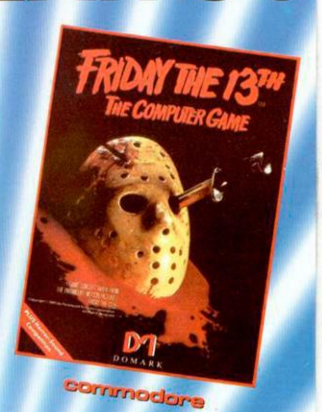
NUESTROS EXITOS



spectrum







commodore



spectrum





spectrum

RECORTA Y ENVIA ESTE CUPON A: SERMA, C/. BRAVO MURILLO, N.º 377.
PISO 3.º A. 28020 MADRID. TELEFONOS: 733 73 11 - 733 74 64

TITULO	PRECIO	CANTIDAD	SISTEMA	NOMBRE Y APELLIDOS:
YABBA DABBA DOO!	2200			
BACK TO SKOOL	2100			
KARATE	1850			DIRECCION:
VIERNES 13	2500			
PANZADROME	2500			

POBLACION: ______ PROVINCIA: ______ CODIGO POSTAL: ______

FORMA DE PAGO: ENVIO TALON BANCARIO ☐ CONTRA REEMBOLSO ☐

MICROHOBBY ESPECIAL

ESPECIAL MICROHOBBY•AÑO II•N.º 2•MARZO 1986

Director Editorial
José I. Gómez-Centurión

Director de Números Especiales
Gabriel Nieto

Director de Microhobby
Domingo Gómez

Redactora Jefe
África Pérez Tolosa

Diseño
José M. Alcoceba

Redactor
Amalio Gómez

Colaboradores
Alejandro Julvez, Marcos Ortiz, Pedro Pérez, Alan Head, David Sopuerta, José M.ª Díaz, José M. Lazo, Jesús Alonso Gallo

Fotografía
Javier Martínez, Carlos Candel

Dibujos
José A. Calvo, F. L. Frontán, J. Igual, Enrique Almendros

Edita
HOBBY PRESS, S.A.

Presidente
María Andrino

Consejero Delegado
José I. Gómez-Centurión

Jefe de Publicidad
Marisa Esteban

Publicidad Barcelona
José Galán Cortés
Tels.: 303 10 22 - 313 71 76

Secretaría de Dirección
Marisa Cogorro

Suscripciones
M.ª Rosa González
M.ª del Mar Calzada

Redacción, Administración y Publicidad
La Granja, 39
Polígono Industrial de Alcobendas
Tel.: 654 32 11
Telex: 49480 HOPR

Dto. Circulación
Carlos Peropadre

Distribución
Coedis, S. A. Valencia, 245
Barcelona

Imprime
ROTEDIC, S. A. Ctra. de Irún, km 12,450 (MADRID)

Fotocomposición
Novocomp, S. A.
Nicolás Morales, 38-40

Fotomecánica
Gráfico Hispano
Rufino González, 32

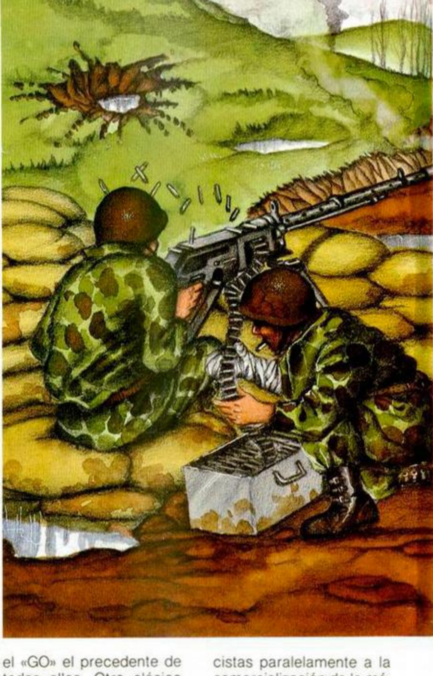
Depósito Legal:
M-36.598-1984

Representante para Argentina, Chile, Uruguay y Paraguay, Cía. Americana de ediciones, S.R.L. Sud América 1.532. Tel.: 21 24 64 1209 BUENOS AIRES (Argentina)

Solicitado control OJD

Juegos de guerra





Gabriel Nieto

Con el «boom» del ordenador resonaron los primeros compases de una danza bélica que anunciaba, a bombo y platillo, la llegada de un nuevo género tan antiguo, sin embargo, como el propio ser humano.

Y así, sin darnos cuenta, ante el asombro de unos, la curiosidad de algunos y el regocijo de otros, surgieron de la máquina los primeros ecos de explosiones procedentes de singulares batallas libradas en un continente computerizado, donde hombre y máquinas luchan por la posesión del imperio cibernético.

El ser humano ha sentido desde siempre un desmedido afán de lucha que han reflejado culturas tras culturas, en un desfile sin par de obstentación bélica. El símbolo del guerrero y la guerra en sí han marcado el sentido de todas las civilizaciones que nos han precedido e incluso, en la nuestra con un cariz mucho más agudizado aún.

Y el ordenador, que es fiel reflejo del mundo en el que nos desenvolvemos, no podía olvidarse de un aspecto tan peculiar y representativo. Y claro está, no lo ha hecho.

Dentro de los juegos de ordenador y afinando más aún, dentro de los juegos de guerra, hay dos caminos claramente diferenciados: el de los arcade del tipo bélico y los clásicos wargames donde es posible reproducir batallas históricas en los mismos escenarios y con las mismas condiciones en las que se produjeron aquéllas.

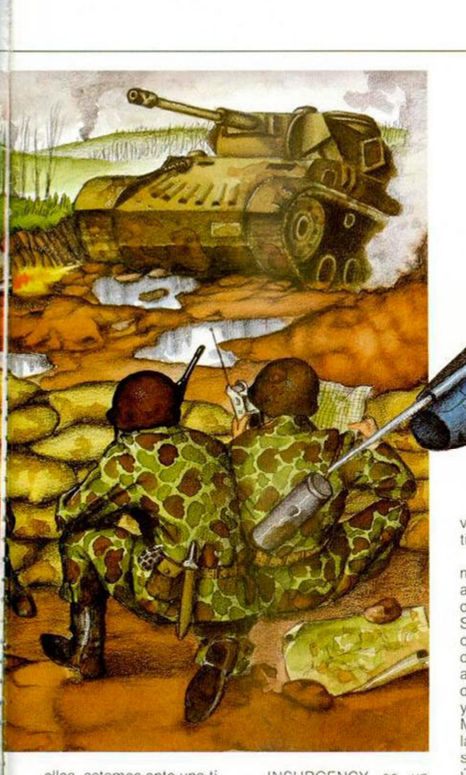
Los wargames

Estos últimos (los wargames) se remiten a una época muy antigua, siendo el «GO» el precedente de todos ellos. Otro clásico del género, aunque a un nivel distinto, es el ajedrez que también ha tenido a ilustres representantes en el Spectrum, ordenador éste que es, al fin y al cabo, el que nos preocupa y ocupa.

Los wargames tienen su cuna y esplendor en el Reino Unido donde existe una gran afición por este tipo de juegos. Por eso no es raro comprobar que todos los programas de estrategia bélica procedan de la Imperial Isla Británica, pródiga en campañas guerreras y conquistas allende los mares.

El Spectrum tiene sus primeros escarceos belicistas paralelamente a la comercialización de la máquina. Un claro exponente de lo que estamos diciendo es «BATLE 1917», un programa pensado para dos jugadores en el que no es posible competir contra el ordenador, pero que nos permite a cambio una libertad de movimientos poco usual en este tipo de juegos. La compañía que se encargó del producto fue *CCS* que seguiría creando nuevos wargames con diferentes temas y escenarios.

«WAR 70» y «WHODUNNIT» son dos claros exponentes del tema bélico con los que la compañía intenta llegar a los aficionados. En el caso del primero de



ellos, estamos ante una típica contienda napoleónica en la que podemos elegir a cualquiera de los dos ejércitos y revivir una emocionante batalla que duró tres días.

Ambientados en un terreno más actual, encontramos dos juegos: AIR DEFENCE y NATO ALERT. El primero, de estrategia aérea y el segundo, sobre un hipotético enfrentamiento entre los dos bloques en el continente europeo.

EAST FRONT nos traslada a la Segunda Guerra Mundial, al frente, donde alemanes y rusos entablaron duros combates por conseguir la supremacía del territorio del Este.

INSURGENCY es un programa con un tema poco usual en los wargames. El conflicto se desarrolla en América Central y está basado en una hipotética revolución que el ejército debe intentar aplastar. Intervienen en la acción factores sociales, consideraciones políticas y aspectos logísticos de todo tipo. Todo ello en 100 campos de batalla distintos.

WAR ZONE es otro de esos productos que no están localizados en ningún escenario real y en el que tenemos que mostrar nuestra capacidad frente a la máquina, en este caso por ejemplo, con el fin de logar el control sobre nueve zonas territoriales distintas.

Uno de los wargames más brillantes es, sin lugar a dudas, ARHEN, un juego de estategia basado en la Segunda Guerra Mundial que nos traslada a Holanda, a un escenario próximo al final de la contienda, donde los acontecimientos y la habilidad del general Montgomery precipitaron la caída de Hitler y el consiguiente deterioro de los últimos cartuchos del ejército alemán. El juego es de lo mejor que se ha realizado en wargames y la crítica así lo reconoció tras su aparición en Inglaterra.

Pero la historia no acaba aquí y recientemente llegaba un nuevo programa de *CCS*, WATERLOO, un claro exponente del género, muy parecido a Arhen (casi idéntico) en el que se habían empleado todos los elementos del anterior wargame, con la salvedad de que en éste, el enfrentamiento se produce en un marco histórico bastante distinto, el que enfrentó al ejército inglés y sus aliados contra las tropas napoleónicas en lo que sería el último intento del emperador en sus ansias por recuperar el poder perdido.

La otra compañía por excelencia especialista en wargames es *Lothlorien* que tiene un amplio historial de creacciones basadas en conflictos bélicos. «PARAS», por ejemplo, está basado en una situación imaginaria en la que dos ejércitos compiten en su afán por conseguir un puente de vital importancia para logar el control de la zona.

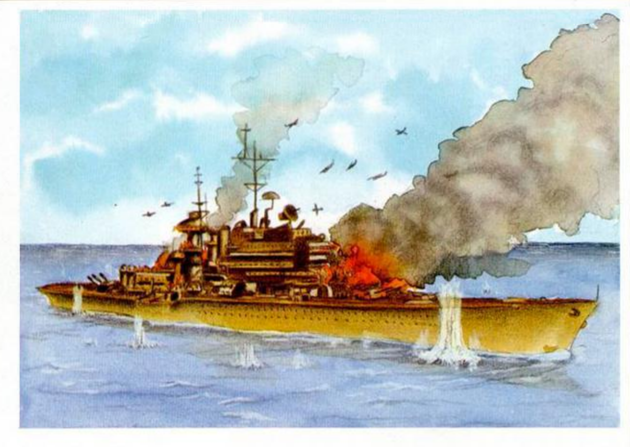
Otro producto de similares características, y la verdad sea dicha, no demasiado brillante, se llama JOHNY REB. Es también de estrategia y reproduce una contienda, la de la Guerra de la Independencia que enfrentó al ejército de la Unión contra los Confederados.

DREADNOUGHTS es una batalla naval que opone a los alemanes contra las fuerzas de la Royal Navy en lo que sería el principio de un conflicto con una gran repercusión en la Segunda Guerra Mundial.

CONFRONTATION es un producto algo distinto a los otros porque tiene continuación en CONFRONTATION SCENARIOS, un programa con cuatro escenarios distintos en los cuales

se desarrolla la acción: Afghanistan, Angola, Sinaí y la invasión germana de 1940. Este juego precedería a otros volúmenes, de una serie que seguramente tenga más ampliaciones.

Paralelamente, intentan el éxito con otros programas más del tipo arcade, pero también ambientados en conflictos bélicos, como es el caso de BATTLEZONE y RED BARON. En este último nos convertiremos durante unos instantes en el temible Barón Rojo luchando por el demonio de los aires al servicio del Kaiser.



Contiendas marítimas

El mar también es un escenario muy propicio para combatir y los chicos de *PPS*, que lo sabían muy bien, decidieron emprender un proyecto ambicioso que se llamaba «BATTLE FOR MIDWAY». Estaba bien desarrollado en todas sus fases. El programador es Alan Steel uno de los hombres que más domina este tema y el escenario, Pearl Harbor, el lugar hacia donde se dirigieron, el 4 de junio, cuatro portaaviones de la armada japonesa al mando del almirante Yamoto.

CCS también se ocupa de la contienda entre nipones y americanos y lanza PACIFIC WAR. Este juego nos sitúa en la Batalla de Guadalcanal donde dirigiremos al ejército americano. Disponemos de aviones de combate, buques de guerra, cruceros y aeroplanos.

ATRAM

Dentro de los wargames hay un claro intento por parte de la compañía *PD* Visual Marketing de crear un producto distinto a todos los otros y que quiere ser algo intermedio entre los juegos de tablero clásicos y los de ordenador. Se trataba de ATRAM y venía con una presentación a todo lujo capaz de impresionar a cualquier aficionado. Unos tableros metálicos, fichas imantadas y todo el encanto del ordenador al servicio de una buena idea que, desgraciadamente, no tuvo la acogida esperada en un principio, el precio seguramente fuera una razón de peso para que así ocurriera.



Los nuevos

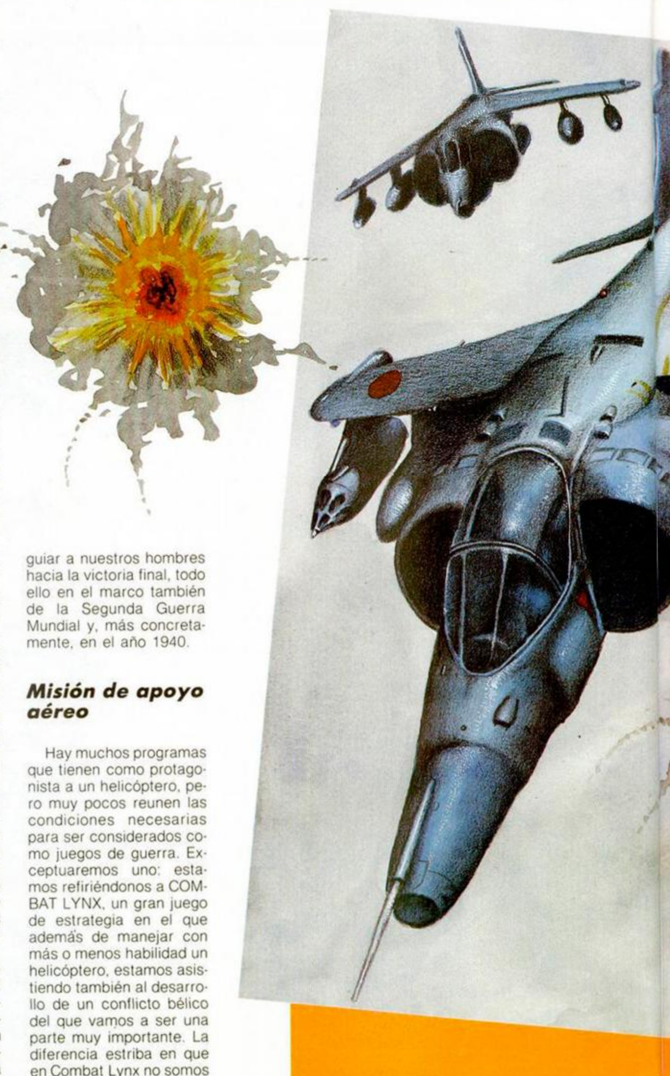
Recientemente, han aparecido dos nuevos juegos bastante interesantes COMBAT ZONE, de la compañía *CRL*, y DESERRATS de *CCS*. Este último basado en la campaña de Africa, donde el ejército inglés y las temibles fuerzas de Romel entablaron una de las batallas más difíciles de cuantas se produjeron durante la Segunda Guerra Mundial. El juego está en la línea de ARHEN y le supera en algunos aspectos. Tiene seis escenarios distintos: la captura de Tobruk, la ofensiva de Battlexade, la Operación Crusader, la batalla de Gazala, el Alamein y la Guerra del Desierto.

Si los wargames han sido prolíferos en el mundo

del Spectrum los juegos de estrategia aérea no lo han sido menos. Recordemos si no títulos legendarios como FHIGTER PILOT. Un juego que, además de ser un magnífico simulador de vuelo, reunía todas las condiciones tácticas necesarias para hacernos creer que estábamos pilotando un avión real, que en cualquier momento podía ser alcanzado por el fuego enemigo.

Desde éste al siguiente hay, sin embargo, un largo paréntesis y es ahora cuando parece haber en la mayoría de los productores de software un claro deseo por inundarnos con juegos de estrategia aérea. NIGHT GUNER, de Digital (la misma del Fighter Pilot) era el único que seguía la tradición hasta la llegada de los nuevos héroes del aire. JUMP JET basado en el avión Harrier y TOMAHAWK en un helicóptero de combate, son dos claros exponentes del género. Pero el auténtico bombazo es un juego de *VS GOLD*, DAN BUSTER, basado en la operación aérea que llevara a cabo el ejército inglés durante la Segunda Guerra Mundial, cuando fue enviado a territorio enemigo el escuadrón 617 de la RAF con el fin de bombardear enclaves de vital importancia para el ejército alemán. Con Dan Busters podremos dirigir una operación aérea asumiendo los papeles de todos los componentes de un avión Lancaster. Sin embargo, no podemos decir que sea un juego puro de estrategia ya que reúne en un sólo programa varios elementos, incluido el arcade.

El último de estos proyectos ha sido SPITFIRE de la compañía de software Mirrorsoft y nos traslada a la cabina de una avión desde el cual y como capitanes de una escuadrilla de la RAF, tendremos que guiar a nuestros hombres hacia la victoria final, todo ello en el marco también de la Segunda Guerra Mundial y, más concretamente, en el año 1940.



Misión de apoyo aéreo

Hay muchos programas que tienen como protagonista a un helicóptero, pero muy pocos reunen las condiciones necesarias para ser considerados como juegos de guerra. Exceptuaremos uno: estamos refiriéndonos a COMBAT LYNX, un gran juego de estrategia en el que además de manejar con más o menos habilidad un helicóptero, estamos asistiendo también al desarrollo de un conflicto bélico del que vamos a ser una parte muy importante. La diferencia estriba en que en Combat Lynx no somos los responsables de cada



uno de los movimientos de nuestro ejército, sino que representamos a una unidad de apoyo que podrá ser decisiva o no en la contienda dependiendo de la habilidad, astucia y visión de juego con las que seamos capaces de reaccionar en el momento oportuno.

Los arcade

US GOLD es una compañía especialmente dedicada a temas belicistas, pero eso sí, siempre desde el terreno más puramente arcade. Uno de los primeros juegos que lanzó en Europa fue precisamente BEACH HEAD, un programa basado en un conflicto bélico, en el que la misión principal es hacer desembarcar a nuestras tropas en las costas del territorio enemigo y llegar hasta el centro neurálgico de sus defensas, un espectacular cañón que debemos destruir.

El siguiente título de US GOLD es RAID OVER MOSCOW que reproduce un hipotético enfrentamiento entre la URSS y los EE.UU.

Una de sus últimas producciones es BEACH HEAD II que a pesar del título no tiene nada que ver con la primera parte. En esta ocasión podemos adoptar el papel de un dictador o dirigir un ejército contra el país que domina éste, teniendo en cuenta que dependerá de la postura que adoptemos el que ataquemos o tengamos que defendernos.

La guerra de las galaxias

Pero no podemos olvidar que las primeras batallas para ordenador se libraron en los confines de las estrellas, donde el jugador podía convertirse casi por arte de magia, en el piloto de una nave espacial cuyo objetivo era llegar al centro de una ciudad en un lejano lugar de la galaxia, como ocurría, por ejemplo, en un juego llamado PENETRATOR. Otras veces, como en el caso de DEFENDER, teníamos que proteger nuestro planeta del ataque de las fuerzas del espacio.

Muchos títulos han ido desfilando delante de nosotros en una explosión orquestada de marcianos, naves espaciales y rayos láser. AD ASTRA, MOON CRESTA, ARCADIA y ASTRO BLASTER son algunos ejemplos representativos de este género. Pero la obra maestra es, sin lugar a dudas, CODENAME MAT, un juego en el que vamos a sistir a una auténtica guerra de las galaxias. Como comandantes de un escuadrón de combate aéreo y desde una nave galáctica, que es un auténtico centro táctico de mando, tendremos que lanzar una ofensiva en toda regla contra el imperio de los Myons. Planos tácticos, innumerables controles de mando y un escenario casi real, hacían de este juego una auténtica epopeya galáctica que, por cierto, tendría continuidad en CODENAME MAT II.

Cuestión de principios

Dicho esto sólo nos queda hacer una breve puntualización: la violencia no es aconsejable desde ningún punto de vista, las guerras siempre son debastadoras y hay quien piensa que los juegos de este tipo no son recomendables. Nosotros estamos de acuerdo en que nunca debería haber guerras de ninguna categoría, pero ójala todas las guerras se libraran en la pantalla de un ordenador.

El presidente

Spectrum 48 K

El objetivo del juego no es otro que dirigir un país (aunque casi por su dificultad mejor diríamos que es aguantar 12 turnos en el poder más que dirigir). Tenemos un territorio delimitado por 2 países vecinos, denominados Lituania y Asbornia, una población inicial (75.000 habitantes) divididos en tres grandes clases sociales con características muy determinadas (clase alta, obreros y agricultores y militares) y unas producciones (trigo, café, azúcar, armamento, tecnología y petróleo). Además, podemos controlar los salarios y pensiones, los impuestos y los precios de venta al exterior e interior.

El presupuesto inicial es de 3.000 millones de points. Este dinero hay que invertirlo ya. El presupuesto viene a ser mayor o menor con respecto a las ventas interiores y exteriores de los impuestos que nos sumemos y los salarios y pensiones que restemos.

Las producciones ya mencionadas antes (trigo, café,...) tienen un tipo de rendimiento cada una (40 por 100, 50 por 100, 90 por 100, valores aleatorios). Lógicamente no rinde la misma cantidad de toneladas un trigo al 90 por 100 que al 10 por 100). Esto originará que luego quizá tengamos que importar algún producto de nuestros vecinos para cubrir las necesidades de la población. Y lo contrario: con un rendimiento elevado la producción excede las necesidades nacionales y podemos exportarlo. De las producciones del sector primario depende la clase baja, agricultores..., es decir que si no se invierte aunque sea lo mínimo, esto sentará muy mal a la masa obrera, al no encontrar trabajo. De un modo más indirecto, si no se producen alimentos habrá hambre y esto no sólo lo notará la clase baja sino todas las demás. La clase alta controla la industria (armamento, tecnología, petróleo).

Países vecinos. Mediante la opción 3 podemos ver las deudas con cada uno, pagarlas si queremos, ver su nivel económico, y cómo van nuestras relaciones. Las deudas se producen cuando necesitamos comprarles algo para subsistir y no hay dinero. Estas tienen un límite. Las relaciones irán mejorando con el comercio y el trato, y empeoran de modo excesivo si aumentamos en armamento a ritmo excesivo, creyéndose éstos que vamos a invadir su territorio. Cuando no quieren venderte algo aparece en pantalla un mensaje tal que éste: «Por razones políticas nos vemos obligados a no vendérselo.»

Cuando damos a la opción 2, continuar, después de haber modificado y/o establecido los nuevos impuestos, sueldos, inversiones, es cuando tenemos ocasión de ver las cosechas obtenidas y lo que hace falta para la población, pudiendo vender y comprar con Lituania y Asbornia.

Más tarde podemos observar varias gráficas. La primera es del estado de ánimo del pueblo (si está conforme, normal, disgustado, agitado, o muy agitado). La siguiente nos muestra el número de habitantes actual y en anteriores jugadas. Si todo va normal la población crecerá y si ha habido hambre se habrá producido un bajón. Luego aparecerá otra de las producciones y lo que se necesita para ir puliendo las inversiones. También podemos ver los titulares de prensa si ha ocurrido algo fuera de lo normal en la jugada. Estos comentarios periodísticos nos darán la pauta a seguir en nuestro mandato: relaciones diplomáticas, impuestos, quema de cosechas...

Al final de 12 turnos el juego habrá acabado, sucediéndose unas elecciones que nos darán la puntuación obtenida en votos, introduciendo el nombre si hemos batido el récord.

Las elecciones se presentan difíciles. Habrá que competir contra los partidos más representantes de las clases sociales: Clases altas: Pérez Puig (capitalistas unidos), Peter Green (del partido verde), David Croket (convención agrícola).

Algunas consideraciones finales: No conviene tener un ritmo muy elevado en la producción de armamento, lo normal es ir creciendo a un ritmo por jugada de 5 por 100 y con respecto de la contaminación tampoco debe aumentar más por jugada que un 3 por 100 (si es menos, mejor).

Si invertimos mucho en industrias hay un mayor índice de contaminación perjudicando así las producciones agrícolas. Conviene que todos estén conformes con la política llevada a cabo, no sólo porque al final votan más, sino porque se puede producir un caos. Si los obretos se excitan mucho van a la huelga general. Si se enfadan excesivamente los «capitalistas» cierran las fábricas y si son los militares dan un golpe de estado.

Instrucciones para grabar

Primero debemos teclear el listado número 1. Dicho programa realiza los gráficos que

se va a utilizar en el programa, creando un juego de caracteres. Para comprobar si el juego de caracteres está completamente bien basta con colocar un STOP delante del LOAD de la línea 8131, después hacer RUN y cuando aparezca el error 9 de STOP, teclear todas las letras mayúsculas y nos irá apareciendo un juego de caracteres más anchos que los normales del Spectrum.

Para grabar el listado 1, teclear SAVE «Presidente» LINE 8000.

El listado 2 es el programa principal y para grabarlo utilizaremos SAVE «Presi» LINE 100.

Hacer la grabación del listado 1 delante de listado 2.

LISTADO 1

```
   1 DATA 124,254,238,254,254,23
8,238,0,254,254,236,248,236,254,
254,0,126,252,224,224,224,252,12
6,0,252
   2 DATA 254,238,238,238,254,25
2,0,254,252,224,248,224,252,254,
0,254,252,224,248,224,224,224,0,
126,254
   3 DATA 224,238,230,254,126,0,
230,230,230,254,230,230,230,0,12
4,56,56,56,56,56,124,0,254,254,5
6
   4 DATA 56,56,248,248,0,230,23
6,252,254,230,230,230,0,224,224,
224,224,224,254,254,0,198,238,25
4,254
   5 DATA 238,238,238,0,230,230,
246,254,238,230,230,0,124,254,23
8,238,238,254,124,0,252,254,238,
254,252
   6 DATA 224,224,0,124,254,198,
198,214,254,124,0,252,254,238,25
4,252,238,230,0,124,254,224,60,1
4,126
   7 DATA 252,0,254,254,56,56,56
,56,56,0,238,238,238,238,238,254
,124,0
  11 DATA 238,238,238,238,238,12
4,56,0,238,238,238,254,254,238,1
98,0,198,238,124,56,124,238,198,
0,238
  12 DATA 238,108,56,56,56,56,0,
254,252,200,16,38,126,254,0,124,
254,206,214,230,254,124,0,28,60
  13 DATA 124,28,28,28,62,0,124,
206,14,124,192,254,254,0,252,254
,14,60,14,254,252,0,60,124,220
  14 DATA 158,254,28,28,0,254,25
4,224,252,14,206,252,0,60,126,96
,124,102,126,60,0,254,254,6,60
  15 DATA 24,48,96,0,254,238,238
,124,238,238,254,0,60,126,102,12
6,6,126,60,0
  17 DATA 254,0,246,254,238,230,
230,0
8000 PAPER 0: BORDER 0: CLS
8010 CLEAR 63999: POKE 23607,249
8100 FOR X=15616 TO 16383: POKE
X+48384,PEEK X: NEXT X
8101 INK 3: PLOT 0,96: DRAW 255,
0: DRAW 0,-33: DRAW -255,0: DRAW
 0,33
8102 INK 3: PLOT 0,97: DRAW 255,
0: DRAW 0,-35: DRAW -255,0: DRAW
 0,35
8105 FOR X=64264 TO 64471: GO SU
B 9000: READ Y: POKE X,Y: NEXT X
8110 FOR X=64128 TO 64207: READ
Y: POKE X,Y: GO SUB 9000: NEXT X
8120 FOR I=0 TO 7: READ X: POKE
USR "N"+I,X: NEXT I
8130 FOR I=0 TO 4: PRINT AT 1,7;
 FLASH 1; PAPER I; INK 9;"PULSE
PLAY": PAUSE 35: NEXT I
8131 PAPER 0: INK 0: PRINT AT 0,
10;: LOAD ""

8140 STOP
9000 REM ---
9010 RETURN
9999 POKE 23606,0: POKE 23607,60
```

LISTADO 2

```
 100 REM
Simulador de control de un pais.
        SENTADO EN LA PRESIDENCIA
 120 REM TITULO POPULAR:
     (sentado en la poltrona)
 130 REM
     ---BY PABLO I SANCHEZ---

 135 LET RE=0: LET R$="RUFUS EL
CONQUISTADOR"
```



```
 140 POKE USR "A"+0,255: POKE US
R "A"+1,129: POKE USR "A"+2,129:
 POKE USR "A"+3,129: POKE USR "A
"+4,129: POKE USR "A"+5,129: POK
E USR "A"+6,129: POKE USR "A"+7,
255
 900 RANDOMIZE

1900 POKE 23658,8
1910 DIM F(3)
1920 DIM G(3)
1930 DIM H(3)
1940 DIM A$(2,8)
1950 DIM A(2)
1960 DIM B(2)
1970 DIM D(2)
1980 DIM E$(2,6)
1990 DIM C(6)
2000 DIM P$(6,10)
2005 DIM P(6)
2010 DIM V(6)
2020 DIM T(6)
2025 DIM O(12): LET O(1)=75000
2030 LET SC=0: LET JUG=1: LET PS
=3000: LET OB=2: LET OBP=0: LET
HB=0: LET MIL=2: LET CAL=2: LET
SU=3: LET IM=3: LET PR=3: LET PA
=0: LET HUG=0: LET CIERR=0: LET
PO=75000: LET TA=0: LET C=0
2040 LET P$(1)="TRIGO": LET P$(2
)="CAFE": LET P$(3)="AZUCAR": LE
T P$(4)="ARMAMENTO": LET P$(5)="
ACERO": LET P$(6)="PETROLEO"
2045 FOR I=1 TO 3: LET F(I)=2: L
ET G(I)=2: LET H(I)=2: NEXT I
2050 FOR I=1 TO 6: LET V(I)=0: L
ET P(I)=INT (RND*100): LET C(I)=
V(I)*P(I): NEXT I
2060 LET A$(1)="LITUANIA": LET A
$(2)="ASBORNIA": LET A(1)=3: LET
 A(2)=3: LET B(1)=0: LET B(2)=0:
 LET D(1)=INT (RND*30)+30: LET D
(2)=INT (RND*30)+30
2070 FOR I=1 TO 2: FOR A=1 TO 6:
 LET E$(I,A)=STR$ (INT (RND*2)):
 NEXT A: NEXT I
3000 PAPER 1: BORDER 1: INK 7: C
LS
3004 LET KURSOR=4
3005 GO SUB 9860: PRINT INK 6; B
RIGHT 1;AT 0,3;"SENTADO EN LA PR
ESIDENCIA"
3010 GO SUB 9800
3030 PRINT BRIGHT 1; PAPER 1; IN
K 6;AT 4,0;" 1)VER SITUACION GEN
ERAL"'" 2)CONTINUAR "'" 3)PAISES
 VECINOS"'" 4)IMPUESTOS,SUELDOS,
PRECIOS.."'" 5)PRODUCCION"'" 6)D
IMISION"'" 7)SALVAR PROGRAMA"
3035 PLOT 0,0: DRAW 255,0: DRAW
0,64: DRAW -255,0: DRAW 0,-64
3040 INK 7: LET MENS=1: LET M$="
USE CURSORES PARA SELECCIONAR"
3050 PRINT AT KURSOR,0; INVERSE
1;">": PRINT AT 12,1; PAPER 1; I
NK 6;M$: PRINT AT 12,MENS; PAPER
 1; INK 6; BRIGHT 1;M$(MENS TO M
ENS)
3060 IF INKEY$="6" OR INKEY$=CHR
$ (10) AND KURSOR<10 THEN PRINT
AT KURSOR,0; PAPER 1;" ": LET KU
RSOR=KURSOR+1
3070 IF INKEY$="7" OR INKEY$=CHR
$ (11) AND KURSOR>4 THEN PRINT A
T KURSOR,0; PAPER 1;" ": LET KUR
SOR=KURSOR-1
3080 LET MENS=MENS+1: IF MENS=29
 THEN LET MENS=1
3090 IF INKEY$<>"8" AND INKEY$<>
CHR$ 13 THEN GO TO 3050
3095 IF KURSOR=5 THEN GO TO 5000
3096 IF KURSOR=9 THEN GO TO 7100
3097 IF KURSOR=10 THEN GO TO 750
0
3100 GO SUB (3300 AND KURSOR=8)+
(3500 AND KURSOR=4)+(5000 AND KU
RSOR=5)+(3400 AND KURSOR=7)+(320
0 AND KURSOR=6)
3110 GO TO 3005
3200 REM ---PAISES VECINOS-,--
```

```
3205 CLS : PRINT AT 0,8;"PAISES
VECINOS": GO SUB 9800
3210 FOR I=1 TO 2: PRINT AT 10+I
,0; PAPER 0; INK 7;" ";A$(I); PA
PER (2 AND INT A(I)>=4)+(5 AND A
(I)<4);: PRINT (" NORMALES " AND
 INT A(I)=3)+("  MALAS   " AND I
NT A(I)=4)+(" MUY MALAS" AND INT
 A(I)=5)+("  BUENAS  " AND INT A
(I)=2)+("EXCELENTES" AND INT A(I
)<=1);
3211 IF D(I)>100 THEN LET D(I)=1
00
3215 PRINT PAPER 4;"      ";:
PRINT AT 10+I,22; PAPER 4;INT B(
I); PAPER 3;AT 10+I,28;" ";INT (
D(I));" ": NEXT I
3220 PRINT AT 9,0; PAPER 7; INK
0;"   PAIS    RELACIONES DEUDAS
   ";AT 14,0; INK 7; PAPER 3;"A"
; PAPER 1; INK 7;"=SITUACION ECO
NOMICA(de 1 a100)"
3221 PRINT AT 4,1;" 1)PAGAR DEUD
AS A ";A$(1);AT 5,1;" 2)PAGAR DE
UDAS A ";A$(2);AT 6,1;" 3)CONTIN
UAR"
3223 INPUT "OPCION...";OP: IF OP
>3 OR OP<1 THEN GO TO 3223
3225 IF OP=3 THEN GO TO 3298
3228 INPUT "QUE CANTIDAD? ";X: I
F X>B(OP) THEN GO TO 3228
3230 LET PS=PS-X: LET B(OP)=B(OP
)-X: PRINT AT 10+OP,22; PAPER 8;
 INK 7;B(OP);"  "
3298 PAUSE 50: PRINT #1;" PULSE
UNA TECLA SR. PRESIDENTE": PAUSE
 0
3299 GO SUB 9850: RETURN
3300 REM ---PRODUCCION---
3305 CLS : PRINT TAB 7;"PRODUCCI
ON"
3310 GO SUB 9800
3320 FOR I=1 TO 6: PRINT AT I+5,
0; PAPER 0; INK 6;I;")";P$(I);"
";AT I+5,14; PAPER 2; INK 6;" ";
AT I+5,13;P(I);AT I+5,15;"%  ";
PAPER 3; INK 6;AT I+5,18;"
 ";AT I+5,20;V(I): NEXT I
3325 PRINT '' INK 0; INVERSE 1;"
A"; INVERSE 0; INK 6;"=PRODUCTO"
' INK 2; INVERSE 1;"A"; INVERSE
0; INK 6;"=RENDIMIENTO"' INK 3;
INVERSE 1;"A"; INVERSE 0; INK 6;
"=INVERSION"
3327 INK 7: PRINT AT 18,0; BRIGH
T 1;"VA A MODIFICAR LAS INVERSIO
NES?": IF INKEY$="" THEN GO TO 3
327
3328 PRINT AT 18,0;"
                    ": IF INKEY$
<>"S" THEN GO TO 3330
3329 FOR I=1 TO 6: LET PS=PS+V(I
): INPUT (P$(I));"...";V(I): LET
 PS=PS-V(I): PRINT AT 5+I,20; PA
PER 3; INK 7;V(I);"  ": PRINT AT
 4,0; PAPER 3; INK 7;"PRESUPUEST
O ";INT PS;"  ": NEXT I
3335 PAUSE 50: PRINT #1;" PULSE
UNA TECLA SR. PRESIDENTE": PAUSE
 0
3340: FOR I=1 TO 6: LET C(I)=V(I
)*P(I): NEXT I: GO SUB 9850: RET
URN
3400 REM ---PRECIOS, SUELDOS,IMP
UESTOS...---
3410 CLS : PRINT "  PRECIOS,SUEL
DOS,IMPUESTOS..."
3415 GO SUB 9800
3420 INK 7: PRINT '' PAPER (2 AN
D SU<=2)+(1 AND SU>=3);"SUELDOS
";("MUY ALTOS" AND SU=5)+("ALTOS
" AND SU=4)+("NORMALES" AND SU=3
)+("BAJOS" AND SU=2)+("MUY BAJOS
" AND SU=1)
3430 PRINT PAPER (2 AND IM>=4)+(
1 AND IM<=3);"IMPUESTOS ";("MUY
ALTOS" AND IM=5)+("ALTOS" AND IM
=4)+("NORMALES" AND IM=3)+("BAJO
S" AND IM=2)+("MUY BAJOS" AND IM
=1)
```

```
3440 PRINT PAPER (2 AND PR>=4)+(
1 AND PR<=3);"PRECIOS ";("MUY AL
TOS" AND PR=5)+("ALTOS" AND PR=4
)+("NORMALES" AND PR=3)+("BAJOS"
 AND PR=2)+("MUY BAJOS" AND PR=1
)
3450 PRINT '' PAPER 1; INK 7;" 1
) "; PAPER 7; INK 0;"BAJISIMOS"'
 PAPER 1; INK 7;" 2) "; PAPER 7;
 INK 0;"BAJOS"' PAPER 1; INK 7;"
 3) "; PAPER 7; INK 0;"NORMALES"
' PAPER 1; INK 7;" 4) "; PAPER 7
; INK 0;"ALTOS"' PAPER 1; INK 7;
" 5) "; PAPER 7; INK 0;"MUY ALTO
S"
3455 PRINT ''''"INTRODUZCA EL NUM
ERO DE OPCION"
3460 INPUT "NUEVAS PENSIONES Y S
UELDOS? ";SU: IF SU<1 OR SU>5 TH
EN GO TO 3460
3470 INPUT "NUEVOS IMPUESTOS? ";
IM: IF IM<1 OR IM>5 THEN GO TO 3
470
3480 INPUT "NUEVOS PRECIOS? ";PR
: IF PR<1 OR PR>5 THEN GO TO 348
0
3495 PAUSE 50: PRINT #1;" PULSE
UNA TECLA SR. PRESIDENTE": PAUSE
 0: GO SUB 9850: RETURN
3500 REM ---SITUACION GENERAL---
3510 CLS : PRINT TAB 7;"SITUACIO
N GENERAL"
3513 GO SUB 9800
3515 INK 7: PRINT '' PAPER (2 AN
D SU<=2)+(1 AND SU>=3);"SUELDOS
";("MUY ALTOS" AND SU=5)+("ALTOS
" AND SU=4)+("NORMALES" AND SU=3
)+("BAJOS" AND SU=2)+("MUY BAJOS
" AND SU=1)
3517 PRINT PAPER (2 AND IM>=4)+(
1 AND IM<=3);"IMPUESTOS ";("MUY
ALTOS" AND IM=5)+("ALTOS" AND IM
=4)+("NORMALES" AND IM=3)+("BAJO
S" AND IM=2)+("MUY BAJOS" AND IM
=1)
3519 PRINT PAPER (2 AND PR>=4)+(
1 AND PR<=3);"PRECIOS ";("MUY AL
TOS" AND PR=5)+("ALTOS" AND PR=4
)+("NORMALES" AND PR=3)+("BAJOS"
 AND PR=2)+("MUY BAJOS" AND PR=1
)
3525 PRINT ' BRIGHT 1;"POBLACION
 (aprox.): "; INK 6;PO
3527 PRINT ' PAPER 0; INK 7; BRI
GHT 1;" HAMBRE: ";HB;"% "'" CONT
AMINACION: ";INT C;"%"'" NIVEL A
RMAMENTISTA: ";INT TA;"% "
3530 PRINT ' INK 7;"OBREROS: ";
PAPER (2 AND OB>3)+(1 AND OB<=3)
;("CONFORME" AND OB=1)+("EN CALM
A" AND OB=2)+("AGITADA" AND OB=3
)+("DISGUSTADA" AND OB=4)+("MUY
DISGUSTADA" AND OB=5)
3540 PRINT INK 7;"MILITARES: ";
PAPER (2 AND MIL>3)+(1 AND MIL<=
3);("CONFORME" AND MIL=1)+("EN C
ALMA" AND MIL=2)+("AGITADA" AND
MIL=3)+("DISGUSTADA" AND MIL=4)+
("MUY DISGUSTADA" AND MIL=5)
3550 PRINT INK 7;"CLASE ALTA: ";
 PAPER (2 AND CAL>3)+(1 AND CAL<
=3);("CONFORME" AND CAL=1)+("EN
CALMA" AND CAL=2)+("AGITADA" AND
 CAL=3)+("DISGUSTADA" AND CAL=4)
+("MUY DISGUSTADA" AND CAL=5)
3560 PRINT ' PAPER 3;"PRESUPUEST
O: ";INT PS
3565 PRINT ' PAPER 3;"JUGADA: ";J
UG
3580 PAUSE 50: PRINT #1;" PULSE
UNA TECLA SR. PRESIDENTE": PAUSE
 0: GO SUB 9850: RETURN
3600 STOP
5000 REM ---CONTINUAR---
5010 LET TA=TA+((P(4)+V(4)))/500
0
5020 CLS : PRINT AT 0,0;" PRODU
CCIONES SECTOR PRIMARIO": GO SUB
 9800
5030 FOR I=1 TO 3
5040 LET C(I)=P(I)+V(I): LET TN=
C(I): LET NN=-INT (-PO/3000)+100
0
5050 IF I<4 THEN PRINT AT I+3,0;
" "; PAPER 0; INK 7;P$(I); PAPER
 2;TN;" T"; PAPER 3;" ";NN;" T"
5060 IF I=4 THEN PRINT AT 4,0;"
"; PAPER 0; INK 7;P$(I); PAPER 2
;" ";INT TA;" % "; PAPER 3;" -
- "
5070 IF I=5 OR I=6 THEN PRINT AT
 I,0;" "; PAPER 0; INK 7;P$(I);
PAPER 2;TN;" T"; PAPER 3;" ";NN;
" T"
5080 PRINT AT 10,0;"1) COMPRAR A
 ";A$(1)'"2) COMPRAR A ";A$(2)'"
3) VENDER A ";A$(1)'"4) VENDER A
 ";A$(2)'"5) CONTINUAR";AT 20,0;
 PAPER 2;"A"; PAPER 1;"=PRODUCCI
ON OBTENIDA"' PAPER 3;"A"; PAPER
 1;"=PRODUCCION MINIMA POBLACION
AL"
5090 PRINT AT 16,0; INK 1;"■"; P
APER 3; INK 7;"PRESUPUESTO: ";INT
 PS;" "
5100 INPUT "OPCION...?(1-5) ";OP
: LET OP2=(1 AND OP=1)+(1 AND OP
=3)+(2 AND OP=2)+(2 AND OP=4): I
F OP<1 OR OP>5 THEN GO TO 5100
5110 GO TO (5120 AND OP=1)+(5120
 AND OP=2)+(5170 AND OP=3)+(5170
 AND OP=4)+(5240 AND OP=5)
5120 REM ---COMPRAR---
5130 IF E$(OP2,I)="0" THEN PRINT
 AT 18,0;A$(OP2);" NO TIENE SUFI
ENTE ";P$(I): GO TO 5080
5140 IF A(OP2)=5 OR B(OP2)>900 T
HEN PRINT AT 18,0;"POR RAZONES P
OLITICAS NOS VEMOS": PAUSE 100:
PRINT AT 18,0;"OBLIGADOS A NO PO
DER VENDERSELO": LET A(OP2)=A(OP
2)+1: GO TO 5080
5145 IF I=4 THEN LET TA=TA+5: LE
T PS=PS-1000: PRINT AT 18,0;"O.K
. ARMAMENTO COMPRADO ":
BEEP .1,INT (RND*10): PAUSE 30
5150 PRINT AT 18,0;"O.K.
                  ": LET PS=PS
+(TN-NN)/50: LET BB=TN: LET TN=N
N: LET C(I)=NN: LET A(OP2)=A(OP2
)-.5: LET D(OP2)=D(OP2)+NN/2500:
 IF PS<0 THEN LET PS=0: LET B(OP
2)=B(OP2)-(BB-NN)/50: PRINT AT 1
8,4;",+DEUDAS CON ";A$(OP2)
5160 GO TO 5080
5170 REM ---VENDER---
5180 IF I=4 THEN LET TA=TA-5: LE
T PS=PS+1000: PRINT AT 18,0;"O.K
. ARMAMENTO VENDIDO ":
BEEP .1,INT (RND+10): PAUSE 30
5190 IF E$(OP2,I)="1" THEN PRINT
 AT 18,0;A$(OP2);" TIENE SUFIENT
E ";P$(I): GO TO 5080
5200 IF (D(OP2)<30 AND PR>3) OR
(D(OP2)<75 AND PR>3) THEN PRINT
AT 18,0;"DEMASIADO CARO PARA ";A
$(OP2);" ": GO TO 5080
5210 IF A(OP2)=5 OR B(OP2)>900 T
HEN PRINT AT 18,0;"POR RAZONES P
OLITICAS NOS VEMOS": PAUSE 100:
PRINT AT 18,0;"OBLIGADOS A NO PO
DER COMPRARSELO": LET A(OP2)=A(O
P2)+1: GO TO 50800
5220 LET PS=PS+(TN-NN)/51-PR: LE
T TN=NN: LET D(OP2)=D(OP2)-0.5:
LET C(I)=NN
5230 PRINT AT 18,0;"O.K.
                  ": GO TO 508
0
5240 IF SC THEN RETURN
5250 NEXT I
5260 PAUSE 50: PRINT #1;" PULSE
UNA TECLA SR. PRESIDENTE": PAUSE
 0
5270 GO SUB 9850
5280 PRINT AT 0,7;"SECTOR INDUST
RIAL": GO SUB 9800
5290 PRINT AT 4,1; PAPER 8;P$(4)
;" ";INT TA;"%";" --
5300 LET SC=1: LET I=4: LET NN=T
N-15: GO SUB 5040
5310 LET I=5: GO SUB 5040
5320 LET I=6: GO SUB 5040: LET S
C=0
5330 REM -MODIFICAR VARIABLES---
5340 FOR I=1 TO 2: LET A(I)=A(I)
+(1 AND TA>50)+(0 AND TA<50)+(1
AND B(I)>900): IF A(I)>5 THEN LE
T A(I)=5
5350 IF A(I)<1 THEN LET A(I)=1
5360 NEXT I
5370 LET JUG=JUG+1: LET ANT=CAL:
 LET CAL=0: FOR I=4 TO 6: LET CA
L=CAL+(1 AND V(I)>=500 AND V(I)<
700)+(2 AND V(I)>=700)+(-1 AND V
(I)<500 AND V(I)>200)+(-2 AND V(
I)<=200): NEXT I
5380 LET CAL=CAL+(1 AND PR=4)+(2
 AND PR=5)+(-1 AND PR=2)+(-2 AND
 PR=1)
5390 LET CAL=(6-CAL+ANT)/2
5400 LET CAL=INT (CAL)
5410 IF CAL<1 THEN LET CAL=1
5420 IF CAL>5 THEN LET CAL=5
5430 LET F(1)=F(2): LET F(2)=F(3
): LET F(3)=CAL
5440 IF (F(3)+F(2)+F(1))/3>4.5 T
HEN LET CIERR=1
5445 IF (H(3)+H(2)+H(1))/3>4.5 T
HEN GO TO 9600
5450 LET C=C+(V(4)+V(5)+V(6))/90
0
5460 LET H$=(STR$ ((C(1)+C(2)+C(
3))/PO)): LET H$=H$+"0000"
5470 LET HB=(0 AND VAL H$>1)+(10
0-VAL (H$(3 TO 4)) AND VAL H$<1)
5480 LET ANT=OB: LET OB=0: LET O
B=OB+(-1 AND HB=0)+INT (HB/10)+(
-1 AND PR=2)+(-2 AND PR=1)+(1 AN
D PR=4)+(2 AND PR=5)+(-1 AND IM=
2)+(-2 AND IM=1)+(1 AND IM=4)+(2
 AND IM=5)+(-1 AND SU=4)+(-2 AND
 SU=5)+(1 AND SU=2)+(2 AND SU=1)
+TA/10+C/10
5485 LET OB=INT (OB+ANT/2)
5490 IF OB<1 THEN LET OB=1
5500 IF OB>5 THEN LET OB=5
5505 LET G(1)=G(2): LET G(2)=G(3
): LET G(3)=OB
5510 IF (G(1)+G(2)+G(3))/3>=4.6
THEN LET HUG=1
5520 LET ANT=MIL: LET MIL=0: LET
 MIL=MIL+(HB/10)-(TA-JUG*3): LET
 MIL=INT ((MIL+ANT+OB+CAL)/4): I
F MIL>5 THEN LET MIL=5
5530 IF MIL<1 THEN LET MIL=1
5540 LET H(1)=H(2): LET H(2)=H(3
): LET H(3)=MIL
5550 LET PO=(PO-(((PO/100)*HB)/2
) AND HB<>0)+(PO+((PO/100)*((6-P
R)+(6-SU)-1)) AND HB=0): LET PO=
INT (PO)
5560 LET PS=PS+(IM-SU)*1000: LET
 PS=PS+((C(1)+C(2)+C(3))*PR)/80
5570 REM ---ESTUDIO DEL AÑO---
5575 CLS
5580 PRINT TAB 9;"ESTUDIO GLOBAL
"
5590 GO SUB 9800
5600 FOR I=1 TO 6: PRINT AT 9-I,
1;(CHR$ (64+I) AND I<>6)+(CHR$ 3
2 AND I=6);" ________";AT 9-I,11;
(CHR$ (64+I) AND I<>6)+(CHR$ 32
AND I=6);" ________";AT 9-I,21;(C
HR$ (64+I) AND I<>6)+(CHR$ 32 AN
D I=6);" ________": NEXT I
5610 PLOT 23,104: DRAW 0,40: PLO
T 103,104: DRAW 0,40: PLOT 183,1
04: DRAW 0,40
5620 FOR I=1 TO 3
5630 FOR A=1 TO 5
5640 PRINT AT 9-A,I*2+2; INK 3;(
"■" AND F(I)>=A);AT 9-A,I*2+12;
INK 4;("■" AND G(I)>=A);AT 9-A,I
*2+22; INK 5;("■" AND H(I)>=A)
5650 NEXT A
5660 NEXT I
5670 PRINT AT 10,4;"CLASE";AT 11
,4;"ALTA";AT 10,13;"OBREROS";AT
10,23;"MILITARES"
5680 PRINT '''"A=CONFORME;B=NORMA
L;C=DISGUSTADAD=AGITADA;E=MUY AG
ITADA"
5690 PRINT ''"ESTADO DE ANIMO DE
L PUEBLO."'"LA TERCERA COLUMNA M
UESTRA EL    MOMENTO ACTUAL."
5700 PAUSE 50: PRINT #1;" PULSE
UNA TECLA SR. PRESIDENTE": PAUSE
 0: GO SUB 9850
5710 CLS : PRINT AT 0,6;"GRAFICA
 POBLACIONAL": GO SUB 9800
5720 PLOT 55,16: DRAW 239-55,0:
PLOT 55,16: DRAW 0,134
5730 PRINT AT 21,8;"1 2 3 4 5 6
 7 8 9 0 1 2 "
5740 FOR I=3 TO 19: PRINT AT I,7
;"________________________": NEX
T I
5750 LET O(JUG)=PO: FOR I=1 TO J
UG: FOR A=1 TO INT (O(I)/7000):
PRINT AT 20-A,I*2+6;"■": NEXT A:
 NEXT I
5760 LET X$="POBLACION ACTUAL: "
: LET Y$="      "+(STR$ INT PO)
+" H   "+(" " AND PO<=9999): FO
R I=1 TO LEN X$: PRINT AT 2+I,2;
 INVERSE 1;X$(I TO I);AT 2+I,3;
INVERSE 1;" ";AT 2+I,4; INVERSE
1;Y$(I TO I): NEXT I
```




```
5770 PAUSE 50: PRINT #1;" PULSE
UNA TECLA SR. PRESIDENTE": PAUSE
 0: CLS
5775 GO TO 6170
5780 IF JUG=12 THEN GO TO 9000
6000 PRINT AT 0,0;TAB 6;"PROXIMA
S INVERSIONES": GO SUB 9800
6010 PRINT AT 3,0; PAPER 3;"PRES
UPUESTO: ";INT PS
6020 FOR I=1 TO 6: PRINT AT 3,0;
 PAPER 3;"PRESUPUESTO: ";INT PS;
PAPER 1;AT I*2+5,0;I;") ";P$(I)(
1 TO 6);
6030 FOR A=1 TO NN/2000: IF A+10
>=32 THEN PRINT AT 4+I*2,30; PAP
ER 2;"+": GO TO 6050
6040 PRINT AT 5+I*2,10+A; PAPER
1; INK 2;"▄"
6050 NEXT A
6060 FOR A=1 TO P(I)+V(I)/2000:
IF A+10>=32 THEN PRINT AT 6+I*2,
31; PAPER 3;"+": GO TO 6080
6070 PRINT AT 6+I*2,10+A; PAPER
1; INK 3;"▄"
6080 NEXT A
6090 NEXT I
6100 PRINT ' PAPER 2;"A"; PAPER
1;"=PRODUCCION QUE SE NECESITA"
```

```
 PAPER 3;"H"; PAPER 1;"=ANTERIOR
 PRODUCCION OBTENIDA"
6110 FOR I=1 TO 6
6120 INPUT (I);") INV.ANT=";(V(I
)),(" "),( P(I)),"% ACTUAL=";V(I
): LET P$=P$-V(I): IF P$<0 THEN
LET P$=P$+V(I): GO TO 6120
6130 PRINT AT 3,0; PAPER 3;"PRES
UPUESTO:";INT P$;"  "
6140 NEXT I
6150 PAUSE 50: PRINT #1;" PULSE
UNA TECLA SR. PRESIDENTE": PAUSE
 0
6160 GO SUB 9850
6165 GO TO 3000
6170 PRINT AT 0,0;" ALGUNOS TITU
LARES DE LA PRENSA"'': GO SUB 98
00
6180 IF HUG=1 THEN LET HUG=0: LE
T RST=8030: GO SUB 7000
6185 IF IM-SU>1 THEN LET RST=813
0: GO SUB 7000
6190 IF P(6)+V(6)<NN AND P(6)+V(
6)>NN/2 THEN LET RST=8070: GO SU
B 7000
6200 IF P(6)+V(6)<NN AND P(6)+V(
6)<NN/2 THEN LET RST=8080: GO SU
B 7000
6210 IF HB>12 THEN LET RST=8060:
 GO SUB 7000
6220 IF G(1)+G(2)+G(3)/3>9 THEN
LET RST=8010: GO SUB 7000
6230 IF G(1)+G(2)+G(3)/3>11 THEN
 LET RST=8020: GO SUB 7000
6240 IF C>3*JUG THEN LET RST=804
0: GO SUB 7000
6250 IF TA>5*JUG THEN LET RST=80
50: GO SUB 7000
6255 IF A(1)>4 OR A(2)>4 THEN LE
T RST=8140: GO SUB 7000
6260 LET RF=INT (RND*3)+1: IF C(
RF)-NN>5000 THEN RESTORE 8100+IN
T (RND*2): READ Z$: PRINT 'Z$'P$
(RF): READ Z$: PRINT Z$
6270 LET RF=INT (RND*3)+1: IF C(
RF)-NN>5000 THEN LET RST=8120: G
O SUB 7000
6999 PAUSE 50: PRINT #1;" PULSE
UNA TECLA SR. PRESIDENTE": PAUSE
 0: GO SUB 9850: GO TO 5780
7000 RESTORE RST: READ Z: FOR I=
1 TO INT (RND*Z)+1: READ Z$: NEX
T I: PRINT 'Z$: RETURN
7100 REM ---DIMISION---
7110 INPUT "ESTA SEGURO? (S/N) "
; LINE X$: IF X$(1)="S" THEN GO
TO 9598
7120 IF X$(1)<>"S" THEN CLS : GO
 TO 3005
7500 REM ---SAVE---
7510 CLS : PRINT AT 0,8;"SALVAR
PROGRAMA": GO SUB 9800
7520 PRINT '''" 1)SALVAR PROG.PA
RA CONTINUAR LA   PARTIDA OTRO D
IA"''" 2)SALVAR PROGRAMA SIN LAS
 VA-     RIABLES(NECESARIAMENTE
TIENES  QUE EMPEZAR DE NUEVO)"
7530 INPUT "OPCION?(1-2) ";OP: I
F OP<1 OR OP>2 THEN GO TO 7530
7540 IF OP=1 THEN SAVE "PRESIDEN
T" LINE 3000:*CLS : GO TO 3005
7550 IF OP=2 THEN CLEAR : SAVE "
PRESIDENT" LINE 1: RUN
```


```
1 TO INT (RND*Z)+1: READ Z$: NEX
T I: PRINT 'Z$: RETURN
7100 REM ---DIMISION---
7110 INPUT "ESTA SEGURO? (S/N) "
; LINE X$: IF X$(1)="S" THEN GO
TO 9598
7120 IF X$(1)<>"S" THEN CLS : GO
 TO 3005
7500 REM ---SAVE---
7510 CLS : PRINT AT 0,8;"SALVAR
PROGRAMA": GO SUB 9800
7520 PRINT '''" 1)SALVAR PROG.PA
RA CONTINUAR LA   PARTIDA OTRO D
IA"''" 2)SALVAR PROGRAMA SIN LAS
 VA-     RIABLES(NECESARIAMENTE
TIENES  QUE EMPEZAR DE NUEVO)"
7530 INPUT "OPCION?(1-2) ";OP: I
F OP<1 OR OP>2 THEN GO TO 7530
7540 IF OP=1 THEN SAVE "PRESIDEN
T" LINE 3000: CLS : GO TO 3005
7550 IF OP=2 THEN CLEAR : SAVE "
PRESIDENT" LINE 1: RUN
```


```
7999 STOP
8000 REM ---TITULARES---
8010 DATA 4,"POR LAS CALLES NUME
ROSOS CARTE-  LES CONTRA LA ACTU
AL POLITICA  DEL GOBIERNO","UNA
MINORIA DE OBREROS FORMA UNAPEQU
EÑA MANIFESTACION","GRUPOS PEQUE
ÑOS DE OBREROS INCI-TAN A LA HUE
LGA","PELIGRO DE HUELGA GENERAL"
8020 DATA 3,"LA HUELGA GENERAL V
A A SER INE- VITABLE","EL PROLET
ARIADO SE AMONTONA EN  MITINES C
ONTRA TU POLITICA","GRANDES MANI
FESTACINES EN LAS   PRINCIPALES
CIUDADES DEL PAIS"
8030 DATA 4,"80% DE LOS TRABAJAD
ORES HAN IDO A LA HUELGA","50% D
E LOS COMERCIOS HA CERRADO","LA
MAYOR CRISIS QUE HA SUFRIDO  NUE
STRO PAIS EN LA HISTORIA","LOS P
IQUETES ESTAN CAUSANDO     DAÑOS
 CONSIDERABLES"

8040 DATA 5,"LOS PRODUCTOS AGRIC
OLAS ESTAN   EMPEORANDO CONSIDER
ABLEMENTE DE- BIDO A LA GRAN CON
TAMINACION","LAS FABRICAS Y EL H
UMO ARRASAN  LAS COSECHAS","EL G
RITO DEL PUEBLO:EXIGIMOS    PODE
R RESPIRAR","LOS VERDES AL PODER
.EL PARTIDO  VERDE GANA VOTOS PA
RA LAS PROXI- MAS ELECCIONES","L
A GENTE ACUDE EN MASA A LOS MI-T
INES DE LA 'CONVENCION AGRICOLA'
"
8050 DATA 4,"GRANDES CAMPAÑAS AN
TI-MILITARES","MANIFESTACIONES A
NTI-ARMAMENTO","LOS MITINES DEL
PARTIDO VERDE  ESTAN SIENDO MUY
 CONCURRIDOS","LA PROPAGANDA ANT
IMILITARIZACIONCORRE POR LAS CAL
LES"
8060 DATA 5,"GRAN CRISIS ALIMENT
ARIA:LA      POBLACION FAMELICA.
","EL HAMBRE SE EXTIENDE POR TOD

O  EL PAIS","CATACLISMO POBLACIO
NAL:EL HAMBRE","LA MORTALIDAD AU
MENTA CADA VEZ  MAS.SITUACION CA
TASTROFICA","EL PROBLEMA INMEDIA
TO,CORTAR EL HAMBRE"
8070 DATA 3,"ESTAMOS EN UN PERIO
DO DE CRISIS INDUSTRIAL DEBIDO A
 LA ESCASED  DE PETROLEO","UN AU
MENTO EN LA OBTENCION DE    PETR
OLEO SOLUCIONARIA EL PROBLE-MA D
EL SECTOR TERCIARIO","INCONVENIE
NTES DEBIDO A LA FALTA DE PETROL
EO"
8080 DATA 3,"INDUSTRIA PARALIZAD
A TOTALMENTE POR LA FALTA DE COM
BUSTIBLE","LAS COMUNICACIONES Y
TRANSPORTESGRAVEMENTE AFECTADOS
POR LA     CRISIS DE PETROLEO","
LA CRISIS PETROLIFERA ESTA PRO-
DUCIENDO GRANDES DAÑOS ECONOMI-
COS A LA INDUSTRIA"
8100 DATA "LA SATURACION DEL MER

CADO DEL "," HA OBLIGADO A NUMER
OSOS PRODUC- TORES A QUEMAR SUS
COSECHAS A   FALTA DE COMPRADORE
S"
8110 DATA "LA SUPERPRODUCCION DE
 "," HA ORIGINADO UN BLOQUEO EN
EL  MERCADO NACIONAL"
8120 DATA 2,"QUEMA DE LAS PRODUC
CIONES AGRI-  COLAS A FALTA DE C
OMPRADORES   ORIGINADA POR LA GR
AN PRODUCCIONOBTENIDA","LOS AGRI
CULTORES SE VEN OBLIGA-  DOS A T
IRAR GRANDES CANTIDADES DE ALIME
NTOS DE UNA SUPERPRO-    DUCCION
 FALTA DE DESTINO"
8130 DATA 4,"EL GOBIERNO INTENTA
 SALIR DE LA POBRE SITUACION ECO
NOMICA SOME-  TIENDO A LA POBLAC
ION A IMPUES- TOS EXCESIVOS","NE
CESITADA BAJADA DE IMPUESTOS","E
L PUEBLO SUFRE LAS CONSECUEN-  C
IAS DE LA INEFICAZ ADMINISTRA- C
ION ECONOMICA PAGANDO ALTOS IM-P
UESTOS","LA POBLACION ACUSA LOS
EFECTOS  DE UNOS ELEVADOS IMPUES
TOS"
8140 DATA 3,"NECESITADA MEJORA D
E LAS RELA-   CIONES DIPLOMATICA
S"," DELICADA SITUACION CON NUES
TROS'VECINOS'.","LA EXTREMA SITU
ACION DIPLOMATICAHA ORIGINADO LA
 RETIRADA DE LAS EMBAJADAS EN NU
ESTRO PAIS"
9000 REM ---ELECCIONES---
9005 GO SUB 9860
9010 FOR I=1 TO 5: FOR Y=1 TO 7:
 BEEP I/26,Y: PRINT AT 1,5; INK
Y; BRIGHT 1;"HA ACABADO TU MANDA
TO": NEXT Y: NEXT I
9012 FOR I=5 TO 25: PRINT AT 2,I
;"-": NEXT I
9015 PRINT '''" -DESPUES DE TUS 3
 AÑOS DE      PRESIDENTE TE HAS
 VUELTO A     PRESENTAR A LAS "
ELECCIONES"..."
9020 PRINT '" ESTOS SON TUS ""CO
MPETIDORES"":"
9030 PRINT '"(1)PETER "; PAPER 4
; BRIGHT 1;"GREEN"; BRIGHT 0; PA
PER 1;" (PARTIDO VERDE)"''"(2)PER
EZ PUIG(CAPITALISTAS        UNI
DOS)"
9040 PRINT '"    LA SUERTE ESTA E
CHADA..."
9050 PRINT 'TAB 8; FLASH 1; PAPE
R 2; INK 6;"PULSA UNA TECLA": PA
USE 0
9055 CLS
9056 PRINT "        E L E C C I O
N E S
"
9057 PLOT 0,0: DRAW 255,0: DRAW
0,138: DRAW -255,0: DRAW 0,-138
9060 LET PR1=5-INT ((F(1)+F(2)+F
(3))/3): LET PR2=5-INT ((G(1)+G(
2)+G(3))/3): LET PR3=5-INT ((H(1
)+H(2)+H(3))/3)
9061 IF PR1=0 THEN LET PR1=1
9062 IF PR2=0 THEN LET PR2=1
9063 IF PR3=0 THEN LET PR3=1
9065 LET ABT=INT (RND*((PO/100)*
10)): LET VOT=PO-ABT: LET VOT3=I
NT (VOT/3)
9070 LET TM=INT ((VOT/PR1+PR2+PR
3)+(INT (RND*500)))
9073 LET PG=INT (VOT3-(VOT3/PR2)
+INT (RND*500)): LET PP=INT (VOT
3-(VOT3/PR1)+(INT (RND*500)))
9080 LET DC=INT (VOT3-(VOT3/PR3)
+(INT (RND*500)))
9088 PRINT AT 6,1; PAPER 2;"TU M
ISMO:"; PAPER 1;"      ";TM;AT 8
,1; PAPER 4;"PETER GREEN:"; PAPE
R 1;"  ";PG;AT 10,1; PAPER 5;"P
EREZ PUIG:"; PAPER 1;"   ";PP;A
T 12,1; PAPER 3;"DAVID CROCKET:"
; PAPER 1;" ";DC
9090 PRINT AT 14,1; PAPER 2;"ABS
TENCIONES:"; PAPER 1;"  ";ABT
9100 PAUSE 50: PRINT #1;" PULSE
UNA TECLA SR. PRESIDENTE": PAUSE
 0
9110 CLS : PRINT AT 0,10;"RECORD
S!!!": GO SUB 9800
9120 PRINT ''; PAPER 2;"RECORD:
";RE; PAPER 1;" "; PAPER 2;"VOTO
S"'' PAPER 3;"RECORDMAN:";R$
9125 PRINT ''"  Y AHORA EN ESTA
PARTIDA..."''
9130 PRINT PAPER 4;"NOMBRE:TU MI
SMO"''"Nº VOTOS:";TM
9140 IF TM>RE THEN PRINT ''"
"; FLASH 1; INK 2; PAPER 6;"HAS
SUPERADO EL RECORD!!"; FLASH 0'
 PAPER 1; INK 7;" INTRODUCE TU N
OMBRE (21CARACT.)": INPUT ">>>";
 LINE R$: LET R$=R$+"
                      ": LET R$=R
$( TO 21): LET RE=TM
9598 INPUT "OTRA PARTIDA? (S/N)"
; LINE Z$: IF Z$(1)="S" THEN GO
TO 140
9599 STOP
9600 REM ---GOLPE DE ESTADO---
9605 GO SUB 9859: PRINT AT 0,0;"
HA HABIDO UN..."''"       "; FL
ASH 1; PAPER 2; INK 6;"GOLPE DE
ESTADO!!!"
9620 FOR X=0 TO .23 STEP .01: BE
EP X,X: BEEP X,X+1: NEXT X: BEEP
 .5,X
9630 PRINT ''"LA SITUACION ERA MU
Y CONFLICTIVA !!!NO PODIA ACABAR
 DE OTRA     FORMA!!!"''" SE HA
INICIADO EL REGIMEN."''"A USTED,
EXPRESIDENTE,SOLO LE    QUEDA UN
A OPORTUNIDAD:
                                D
ESAPARECER!!! Y...
                                P
ULSAR UNA TECLA"
9799 PAUSE 0: GO TO 9598
9800 REM ---LINEAS---
9809 PLOT INK 6;0,161: DRAW INK
6;255,0
9810 PLOT INK 6;0,160: DRAW INK
6;255,0
9815 PLOT INK 2;0,159: DRAW INK
2;255,0
9816 PLOT INK 2;0,158: DRAW INK
2;255,0
9820 RETURN
9850 FOR I=0 TO 21: PRINT AT I,0
; PAPER 1;"
          ": NEXT I: INPUT "":
RETURN
9860 PAPER 1: BORDER 1: INK 7: C
LS : FLASH 0: BRIGHT 0: OVER 0:
RETURN
```

Programas de utilidad para el Spectrum

Para hacer más fácil y completa la utilización de nuestro pequeño ordenador se han creado programas de utilidad que facilitan el manejo de datos, simplifican la tarea de programación o nos ayudan a realizar mejor nuestro trabajo. Movidos por la escasa y a veces nula información sobre los mismos, hemos hecho esta pequeña guía que estamos seguros será de gran utilidad.

Para su realización hemos optado por dividirlos en cinco grupos que contendrán las distintas autoridades, completando el trabajo otra guía con el nombre de la casa distribuidora del programa, su dirección, teléfono y localidad.

Utilidades de gestión

Dentro de este grupo se engloban todos aquellos programas que por su uso se puedan considerar útiles para hacer más fácil las tareas de una oficina o una casa.

ADDRES MANAGER

OCP Sinclair Store

Este programa es idóneo para la creación de una agenda de direcciones de clientes, fichas de alumnos y otras distintas utilidades de archivo.
El programa está preparado para la utilización de una impresora con ochenta columnas y es uno de los mejor preparados para el tratamiento de fichas.

BASE DE DATOS

Investrónica

Especialmente indicado para realizar un archivo de direcciones aunque su uso puede ser muy extenso, desde archivar libros hasta una colección de sellos.
El programa permite la utilización de 10 campos numéricos o alfanuméricos, con una longitud máxima de 25 caracteres en cada campo, menos en el último en el que el número puede llegar a ser de 59 caracteres.
Entre las posibilidades del programa podemos destacar la de poder ordenar el campo que nosotros deseemos preferentemente y en cualquier momento se puede cambiar de orden. Si lo que deseamos es buscar un data dentro del fichero bastará con dar una parte o toda del contenido de un campo para localizarlo rápidamente.
Podemos utilizarlo con impresora y realizar modificaciones de un campo ya introducido.



BASE DE DATOS

ABC Soft

Muy útil a la hora de trabajar con ficheros que necesiten campos muy especiales.
Entre las posibilidades de este programa se encuentra: la impresión de listados, altas, bajas, consulta de fichas, ordenar el fichero, y otras.
Tiene la ventaja de manejar ficheros de varios tipos.

Utilidades

CALC (HOJA DE CALCULO)

Microgesa

De gran utilidad para archivar el precio de distintos materiales y los precios de ventas a varios clientes. Por ejemplo, en caso de necesitar subir los precios de materiales no tendremos más que definir el incremento y todos estos materiales se actualizarán.
El programa se puede usar con microdrive y así ganar en rapidez los datos archivados.

CONTABILIDAD

Microgesa

Programa basado en el Plan General Contable con lo que se puede llevar la contabilidad de una empresa con la única limitación de las cuentas que podemos utilizar.
El programa puede realizar cualquier necesidad de tipo administrativa tal como la de observar el Diario, el Mayor, el Balance de comprobación y de Saldos, y otras aplicaciones.
Con este programa podemos utilizar, aparte de la cinta, el microdrive.

CONTABILIDAD DEL HOGAR

ABC Soft

Ofrece muchas prestaciones en el hogar, como el saber en qué se va el dinero con tanta facilidad o mejor dicho, cómo el carnicero se lleva nuestro dinero.
Con este programa, además de llevar los gastos del hogar y de cualquier necesidad familiar, podemos llevar el estado de la cuenta bancaria.
Si lo deseamos podemos comprobar de modo gráfico los gastos de vestuario de cualquier otra cuenta que deseemos.

CONTABILIDAD DOMESTICA MINITEXTOS FIÇHERO

Investrónica

Paquete con tres programas de gestión.
El programa Contabilidad Personal podemos utilizarlo para llevar una pequeña contabilidad doméstica o de un pequeño negocio sin muchos problemas.
Con el programa Minitextos podemos realizar cualquier operación que se pueda hacer con una máquina de escribir, con la ventaja de no tener ningún error en su confección pues lo podríamos corregir antes de listarlo por impresora. Para la utilización de la impresora el programa está equipado con dos formatos, uno para impresoras de menos de 40 columnas, y otro para más de 40 columnas, al elegir uno de ello al cargar el programa nos pregunta si deseamos 32 ó 64 columnas en pantalla.
Con el programa fichero podremos crear uno a nuestro tamaño, pudiendolo utilizar para multitud de archivos.



CONTABILIDAD DOMESTICA



FICHERO

CONTABILIDAD GENERAL PARA MICRODRIVES

Investrónica

Con él podremos llevar la contabilidad de un pequeño negocio con una mayor rapidez al poder utilizar la unidad de microdrive como almacenamiento de asientos y cuentas que manejemos.
Podemos llevar el Diario de movimientos, introducir asientos, modificar asientos, realizar un Balance de situación, Balance funcional, llevar un total de 100 cuentas y efectuar hasta 800 asientos.
Junto a la cinta, y dentro de su carátula, se adjuntan unas pequeñas instrucciones que son bastantes completas y aclaratorias.

Utilidades



CONTABILIDAD MICRODRIVES


CONTABILIDAD PERSONAL

CONTABILIDAD PERSONAL

Ventamatic

Lleva la contabilidad de nuestra casa o de nuestros gastos, consumos del hogar, gastos de la compra, etc.
Para la utilización del programa debemos de crear una pequeña lista con los números de las cuentas que utilizamos y el nombre de las mismas, tras lo cual estará listo para su uso.
El menú, aparte de lo habitual en un programa de este tipo, permite la búsqueda de una cuenta y su estado actual, pudiendo realizar un balance con todas las cuentas.
Cuando hayamos terminado de trabajar con el programa debemos recordar que necesitaremos realizar una copia de los datos que tenemos archivados en ese momento ya que si no, nos veríamos obligados a teclear cada vez que realizamos un nuevo asiento todo lo anteriormente tecleado.

CONTROL DE STOCKS

Ventamatic

Si poseemos un pequeño negocio de tipo almacén o tienda nos será de gran utilidad saber en cada momento la cantidad de existencias que poseemos, de cualquier material, y con gran facilidad.
En el programa, aparte de llevar la cantidad de materiales que poseemos en ese momento, podemos llevar el precio de compra de un producto y el de venta de dicho producto, así como el proveedor que nos lo suministra.
Una de las grandes ventajas de este programa es el trabajar en el modo de 64 caracteres por columnas, ventaja que viene dada por la opción de inventario de todos los materiales que tengamos detallados con todos los datos expresados en una sola línea, lo que permite visualizar por pantalla un mínimo de veinte materiales.
Debemos teclear, según introducimos productos a cada material, un número o código de cuatro caracteres.
A la hora de realizar la función de facturación notaremos que al terminar la factura nos preguntará el ITE pero como ya sabemos que este impuesto ha desaparecido, lo sustituiremos por el famoso IVA aunque su cambio no es notablemente difícil de realizar con unos mínimos conocimientos de programación en basic.


CONTROL STOCK

CONTROL DE STOCKS

Microbyte

De idénticas características al anterior.
Entre las opciones que incorpora este programa se encuentra la posibilidad de mostrar los productos que se encuentran por debajo del mínimo de stock marcado al crear la ficha, sumario financiero, imprimir los materiales de que disponemos...
Podemos controlar hasta un máximo de 200 materiales en el programa.



CONTROL DE STOCKS

ABC Soft

Al terminar la carga del programa notaremos una diferencia entre los otros programas de este tipo y es que nos dirá el número de fichas que podemos realizar, según sea nuestro Spectrum de 16 ó 48 K.
El manejo de los datos de los productos se realiza a través de códigos, no de los nombres de los productos, por lo que es aconsejable llevar una pequeña guía de códigos utilizados.
Las demás funciones son las propias de los programas de este tipo.

CONTROL DE STOCKS

Microgesa

Este programa tiene la ventaja de poder trabajar con microdrive, controlando los datos a través de éste.
Tiene, por lo demás, las mismas características de los otros programas de este tipo, como son la de ofrecer los materiales que se encuentran por debajo de los mismos, controlar los precios de compra y de venta, etc...

DIRECCIONES

ABC Soft

Como su propio nombre indica, es una utilidad de agenda.
Con él podemos llevar los datos de personas o empresas que necesitemos con alguna frecuencia, localizando fácilmente cada una de ellas.
La cantidad de datos que se pueden utilizar es variable según el Spectrum de que dispongamos, de 16 o de 48 K.
Entre las opciones de que dispone el programa se encuentran las de introducir datos nuevos, corregir, ordenar o borrar.

GESTION DE EFECTOS

Ventamatic

Si necesitamos controlar un número grande de efectos este programa nos será de enorme utilidad.
Otras de ellas es la de poder compilar y listar las fichas que estén introducidas.
Los campos de datos están definidos por el programa así que no podremos definirlos a nuestro gusto, pero estos conceptos no tienen que preocuparnos pues engloban todos los datos que suelen utilizarse (el nombre del libador, el número de factura, la fecha de cobro del efecto y la fecha de facturación).
Con este programa podemos trabajar con un total de hasta 200 facturas.
En la cara dos de la cinta se encuentra un programa-ejemplo para familiarizar al usuario con su manejo.


SITI

SITI

Ventamatic

Detrás de este nombre tan extraño hay un Sistema Integrado de Tratamiento de Información, normalmente llamado archivo.
El programa se puede utilizar para crear fichas de datos diversos al poder definir los campos a nuestro gusto.
Podemos idear tantos campos como deseemos, teniendo en cuenta que cuanto mayor sea la cantidad de campos y su contenido, menos fichas podremos realizar.
Podemos definir también, el formato de impresión de dos maneras diferentes, los caracteres de control del tipo de impresora que poseemos...
La cualidad que aporta como novedad sobre los demás programas de este tipo, es la de incorporar una opción para el listado de etiquetas.

TASWORD

Tasman

Programa pensado exclusivamente para la realización de textos.
Con él podemos realizar un texto con gran facilidad y rapidez pudiendo trabar en formato de 64 columnas, ampliando los caracteres lo preciso.
Si lo deseamos podemos colocar los márgenes de derecha e izquierda, realizar el volcado de un bloque en otra posición distinta a la actual o si lo precisamos, repetir un bloque.
El programa posee un menú de opciones muy completo que nos permite acceder a él en cualquier momento sin deteriorar el texto que estamos realizando en ese momento.
Podemos visualizar el texto con rapidez hacia arriba o abajo de dos maneras diferentes, por líneas o por páginas.

AHORA SÍ

puedes aprender a programar en basic de una vez por todas

¡Solicítalo antes de que se agote!
Hay un número limitado de ejemplares

DEJATE de complicados e incomprensibles sistemas de aprendizaje. Conoce de una vez por todas lo que es el Basic. Es más sencillo de lo que crees, porque ahora tienes algo que estabas esperando hace mucho tiempo: MICROBASIC, una edición corregida y revisada del famoso curso publicado por MICROHOBBY SEMANAL.

MICROBASIC es el libro que te enseñará a ser un experto en programación. Aunque hasta ahora sólo hayas utilizado tu Spectrum para jugar.

MICROBASIC te introducirá, paso a paso, en el Basic. Con ejemplos claros, sencillos y prácticos que irán adquiriendo complejidad según vayas aumentando tu nivel. Hasta llegar a dominarlo por completo.

Aprovecha esta oportunidad, porque ahora si puedes llegar a conocer a fondo tu Spectrum. Ahora, por fin, a tu alcance el método más claro y completo de programación en Basic publicado hasta el momento.

Rafael Prades
MICROBASIC
Por fin un curso práctico y completo de programación para Spectrum

Recorta o copia este cupón y envíalo a HOBBY PRESS, S. A. Apartado de Correos 232. Alcobendas (Madrid)

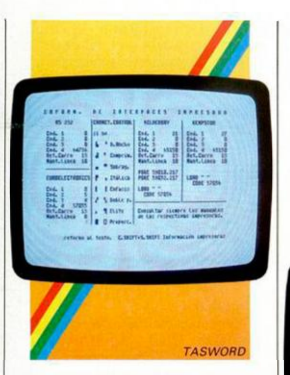
Nombre ______
Apellidos ______
Dirección ______
Localidad ______ Provincia ______
Código Postal ______ Edad ______ Teléfono ______

Deseo recibir en mi domicilio el libro MICROBASIC, al precio de 1.750 ptas. (IVA incluido). El importe lo pagaré:

- ☐ Mediante talón bancario adjunto a nombre de HOBBY PRESS, S. A.
- ☐ Mediante tarjeta de crédito ______
  Número de la tarjeta ______
  Fecha de caducidad de la tarjeta ______
- ☐ Mediante giro postal n.° ______
- ☐ Contra reembolso (supone 75 ptas. de gastos de envío)

Fecha y firma

Utilidades


TASWORD

Otras de las posibilidades son el ensamblado de un párrafo, borrar una línea, ir al principio o al final del texto, centrar textos, buscar una palabra en el texto y cambiar una palabra por otra.
Cuando tengamos que recurrir a la impresora es conveniente ir al menú donde se encuentra la información sobre las mismas.
Podemos, con el procesador de textos, realizar un total de 320 líneas.

VU - FILE

Investrónica

Uno de los programas más conocidos para llevar una agenda o fichero de datos diversos.
Al terminar la carga del programa deberemos realizar la definición de los campos que vamos a utilizar y dar los nombres a los mismos. Si una vez hecho esto deseamos cambiar el nombre de alguno de estos campos, podemos realizarlo sin necesidad de crear de nuevo el fichero.
Una de las funciones que realiza el programa con mayor rapidez es la búsqueda de un dato en cualquier campo de la ficha.
Con la práctica, el programa puede servirnos para la creación de etiquetas y listarlas por impresora.
A la hora de ordenar las fichas podemos realizar el orden por el campo que queramos en cada momento.
También tenemos la ventaja de



V.U.-3D

comprobar cuánta memoria hay ocupada y de cuánta disponemos en cualquier momento.

TRATAMIENTO DE TEXTOS

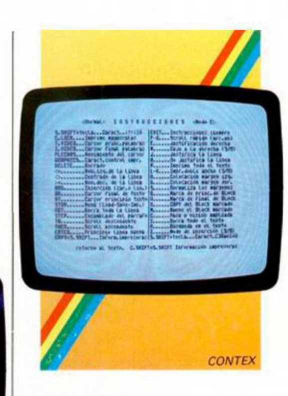
Microgesa

Procesador de textos de idénticas características al anterior, pudiendo realizar las mismas funciones.

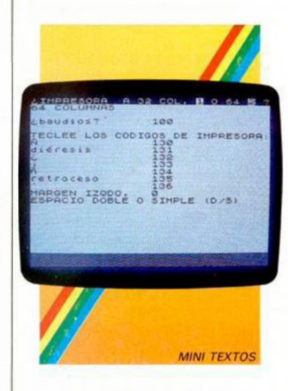
CONTEXT

Ventamatic

Procesador de textos de idénticas características a los anteriores.
La mejor ventaja de este programa es que está preparado para trabajar con cualquier tipo de impresora, con interface serie


CONTEX

RS-232, Centronics, Seikosha y ZX printer.
Con este programa podemos realizar hasta 300 líneas.



MINI TEXTOS

Utilidades

Utilidades de programación

En este apartado vamos a recoger todos los programas que pueden resultarnos de gran ayuda a la hora de realizar los nuestros propios. Entre éstos se pueden incluir los programas de tipo programación en lenguaje máquina y otros lenguajes, desesambladores, compiladores, y programas de ampliación del Basic.

BETA-BASIC

Este programa define más de 30 tokens o comandos que el Spectrum no tiene al encender el ordenador, consiguiendo un Basic comparable con ordenadores más potentes.
Todas las funciones se teclean a través del modo gráfico desapareciendo, por lo tanto, la posibilidad de utilización de los mismos.
Entre las nuevas funciones que disponemos hacemos referencia a las más conocidas del Basic: la función Auto, Renum, Fill, Delete, Edit, Else, Alter, Get, On, On Error y otras menos conocidas, pero de gran utilidad todas ellas.
Además, incorpora un reloj que trabaja por interrupciones para conocer cuánto tiempo hace que cargamos el programa.

TOOLKIT

Microbyte

Programa en código máquina con rutinas para perfeccionar el Basic del Spectrum.
Con él podemos renumerar un programa en Basic, deletrear un bloque de líneas, y otra serie de funciones.
También podemos saber cuánta memoria ocupa el programa Basic.
Es totalmente relocalizable.

COMPILER

Investrónica

Como su nombre indica, es un programa diseñado para compilar programas en Basic y pasarlos a lenguaje máquina, ganando el programa en velocidad.
Tiene la ventaja de no necesitar la presencia en el ordenador del programa una vez compilado.
Podemos compilar un máximo de 8 K aproximadamente, y 10 K de datos adicionales.
El programa compilado no es reubicable y se posiciona en la zona del Basic.
Para ejecutarlo no es necesario teclear el famoso USR, pues funciona con RUN.
Si al realizar la compilación hubiese algún error, el programa nos indicaría que lo admite en la línea en la que se encuentra.
A la hora de compilar hay que programar de manera un poco complicada pues no podemos utilizar todos los comandos del Basic. Pero todo está muy aclarado en el manual de uso del programa.

THE COLT

Hisoft

Si no dominamos el código máquina y queremos realizar programas con mayor rapidez que con el Basic, podemos utilizar el programa Colt para ganar en velocidad.
Realmente el compilador no se suele utilizar para compilar todo el programa en Basic, sino sólo las partes de éste que ralenticen el mismo.
Con este programa podemos compilar hasta un total de 32 K.
En el manual que acompaña la cinta podremos encontrar información sobre el manejo y las limitaciones que tenemos con el programa.

GENS3

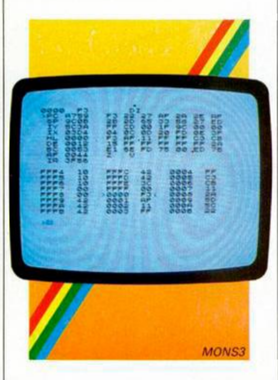
Hisoft

Programa para trabajar en lenguaje ensamblador código máquina, uno de los de mayor calidad que se encuentran en el mercado.
Es reubicable y puede accederse desde el Basic y volver a éste.
Ocupa en memoria unos 9 K lo que nos permite trabajar con algo más de 30 K de memoria.
El manual del programa es muy completo y entre los datos que proporciona se encuentran todos los comandos de manejo que posee, así como todos los nemónicos y para qué se utilizan.
Podemos manejar los datos numéricos en hexadecimal o en



GENS 3

decimal o ambos juntos.
Tenemos la posibilidad de trabajar con etiquetas en los bucles que tengamos definidos no necesitando tener que estar calculando las direcciones en cada creación de un bucle.



MONS3
Hisoft

Esta utilidad permite desensamblar programas escritos en lenguaje código máquina.
El programa trabaja en decimal y hexadecimal, pero en una de ellas sólo.
Las instrucciones son muy completas y en ellas están detallados cada uno de los comandos de funcionamiento del mismo, acompañando un ejemplo de manejo del programa.
Entre las operaciones posibles se encuentran las de listado del programa, insertar datos en el mismo, editar textos, grabar y relocalizar un programa.

MONITOR DESENSAMBLADOR EN CODIGO MAQUINA
Boalox

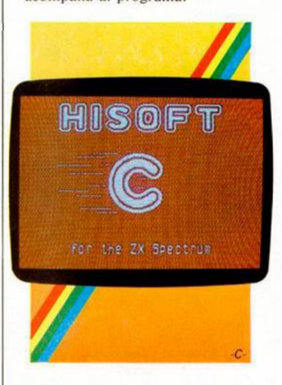
Como su nombre indica, es un programa para poder leer y corregir programas en código máquina.
Entre el menú de opciones que incorpora se encuentra la posibilidad de traducir programas que estén en lenguaje máquina, a través de esta opción aparece en pantalla la dirección de memoria en la que está el puntero y el carácter que contiene; si deseamos, lo podemos corregir.
Este programa podemos también utilizarlo como conversor de hexadecimal a decimal, de decimal a hexadecimal, de binario a hexadecimal y viceversa.
También podemos realizar el volcado de un bloque de bytes en una dirección distinta a la que se encuentra.



DESENSAMBLADOR
Investrónica

Programa monitor de código máquina para la depuración y listado de código máquina.

Existen dos versiones en la cinta dependiendo del tipo de ordenador que utilicemos, si es el 16 ó 48 K.
El manejo del programa está detallado en el manual que acompaña al programa.



-C-
Hisoft

Si deseamos aprender o perfeccionar nuestros conocimientos en lenguaje -C-, lo podremos realizar con este compilador con gran facilidad y con ayuda de un completísimo manual que proporcionan al comprar el programa.
En el manual encontraremos desde los errores de lenguaje hasta rutinas de utilidad a la hora de utilizar dicho lenguaje.

FORTH
Investrónica

Al igual que el -C- el Forth es uno de los lenguajes de programación más conocidos y extendidos.
Con este programa podemos trabajar como con un ordenador

AMSTRAD CPC - 464

AMSTRAD P



ORDENADORES

Esta es la familia de ordenadores personales AMSTRAD. Una familia completa en la que se incluye desde el equipo básico de introducción a la informática hasta el orientado a aplicaciones profesionales. Todo con la filosofía de diseño AMSTRAD que ofrece ordenadores compactos, listos para funcionar sin cableados engorrosos ni necesidad de adquirir periféricos **-con un solo cable a la red-** e incluyendo paquetes de programas de obsequio.

Todos con una tecnología contrastada y fiable basáda en el microprocesador Z 80 A, en el Sistema Operativo CP/M - el más extendido para ordenadores de 8 bits- y en una electrónica depurada y con un riguroso control de calidad.

Todos con una extensa biblioteca de programas que se incrementa día a día con títulos para todos los gustos y necesidades.

Todos con una asistencia técnica rápida y eficaz que AMSTRAD ESPAÑA garantiza exclusivamente a los equipos adquiridos a través de su Red Oficial de Distribuidores y acompañados de la Tarjeta de Garantía de AMSTRAD ESPAÑA.

Todos a unos precios increibles que no admiten comparación con los de cualquier otro ordenador personal de sus características y prestaciones.

AMSTRAD CPC 464.

• Microprocesador Z 80 A• 64K RAM• 32K ROM• Teclado profesional con 32 teclas programables. Sonido estéreo con 3 canales y 8 octavas. Resolución de hasta 640 x 200 puntos. •Texto de 20, 40 y 80 columnas. 27 colores. Conectores multiuso, Centronics, joystick etc... Magnetófono incorporado.

TODO POR: 59.000 pts. (monitor verde)
90.000 pts. (monitor color)

EL SUMINISTRO INCLUYE:

- LIBRO "Guía de Referencia del Programador"
- Manual en castellano
- **8 programas de obsequio en cassette** ("Animal, Vegetal y Mineral", "Amsdraw", "Plaga Galáctica", "Fruit Machine", "Admiral Graph Spee", "Amsword", "El Laberinto del Sultán", "OH. Mummy")

•
4
T
(
c
6
c
m
U
in
T

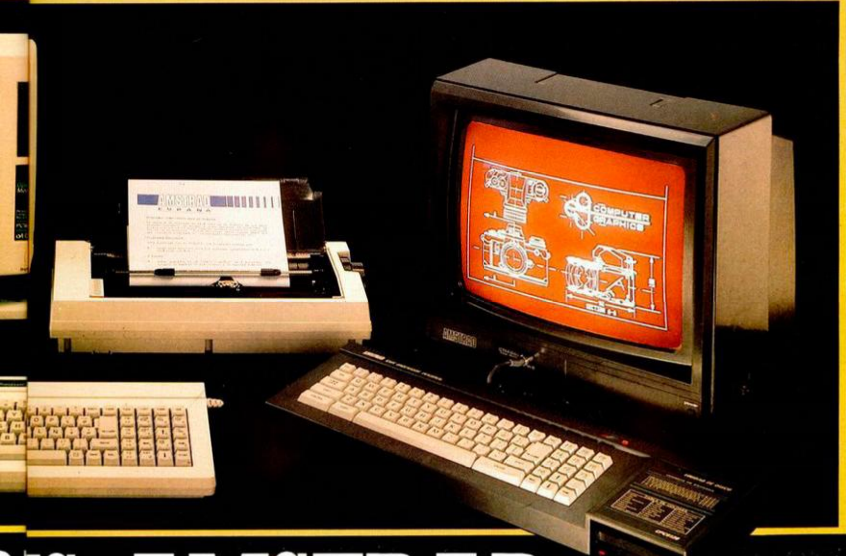
E
•

•

•
•
(

D PCW - 8256

AMSTRAD CPC - 6128



RES AMSTRAD ¡¡Increible!!

AMSTRAD CPC 6128.

• Microprocesador Z 80 A• 128 K RAM• 48K ROM (con BASIC Y AMSDOS) • Teclado profesional de 74 teclas (32 programables). Sonido estéreo con 3 canales y 8 octavas. Resolución de hasta 640 x 200 puntos. Texto de 20, 40 y 80 columnas. 27 colores. Conectores multiuso, Centronics, Joystick, etc... Unidad de disco (3", 180K por cara) incorporados.

TODO POR: 99.900 pts. (monitor verde)
127.900 pts. (monitor color)

EL SUMINISTRO INCLUYE:

- Disco con Sistema Operativo CP/M 2.2 y lenguaje DR. LOGO
- Disco con Sistema Operativo CP/M Plus y Utilidades.
- Manual en castellano
- **Disco con 6 programas de obsequio** ("Base de Datos", "Proceso de Textos I", "Ramdom Files", "Diseñador de Gráficos", "Puzzle", "Animal, Vegetal y Mineral")

AMSTRAD PCW 8256.

• UNIDAD CENTRAL con microprocesador de Z 80 A, 256K RAM y teclado profesional de 82 teclas (ñ, acento, etc...). PANTALLA DE ALTA RESOLUCION con 90 columnas por 32 líneas de texto. UNIDAD DE DISCO de 3" y 180K por cara. IMPRESORA de tracción/fricción con alineación automática de papel.

TODO POR: 129.900 pts.

EL SUMINISTRO INCLUYE: Procesador de textos LocoScript (en castellano). Sistema operativo CP/M Plus. Mallard BASIC con sistema JETSAM (ficheros indexados). Lenguaje DR. LOGO. Manuales en castellano.

NOTA: Es muy importante verificar la garantía del aparato ya que sólo **AMSTRAD ESPAÑA** puede garantizarle la ordenada reparación y sobre todo materiales de repuesto oficiales (Monitor, ordenador, cassette o unidades de discos).

AMSTRAD ESPAÑA

Avda. del Mediterráneo, 9. Tels. 433 45 48 - 433 48 76. 28007 MADRID

Delegación Cataluña: Tarragona, 110 - Tel. 325 10 58. 08015 BARCELONA

Utilidades

que pudiese hacerlo con este lenguaje.
Si deseamos aprenderlo nos será de gran utilidad su utilización.
El Forth es un lenguaje de fácil aprendizaje que al ser compilado por el programa será convertido en código máquina, ganando en velocidad de ejecución.

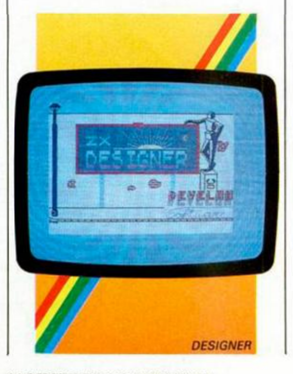
Utilidades gráficas

En este apartado vamos a detallar los programas que nos ayudarán a crear pantallas de presentación, así como las que podemos utilizar para la creación de juegos de caracteres o de UDG.

DESIGNER

ABC Soft

En esta utilidad no encontraremos muchos problemas a la hora de manejarlo.
Entre las peculiaridades del programa se encuentra la de poder aumentar una parte del dibujo de tamaño para poder trabajar con mayor precisión a la hora del detalle.
Podemos también trabajar con impresora y realizar copy de pantalla.
Otras de sus cualidades son las de sombreado de una figura, cuadricular la pantalla, escribir textos, borrar la última operación realizada, crear o reformar caracteres y cambiar el banco de UDGs.



DESIGNER


DRAW

DRAW

Con este programa podremos realizar pantallas de presentación y retocarlas.
Entre las funciones que incorpora se encuentran la de trazar una línea desde un punto fijado con otro entrado después.
Si deseamos aumentar el trazo de dibujo podemos realizarlo de 1 punto hasta 255.
Fill bastante rápido y perfecto, no dejando ninguna parte de la figura sin rellenar.
Gran facilidad de manejo en el uso de las funciones, y posibilidad de colocar textos en cualquier posición de pantalla al tamaño que deseemos.

LEONARDO

Creative Software

En este programa tenemos la posibilidad de trabajar aparte del teclado, con distintos joysticks así como definir el salto que deseamos realizar al trazar una línea, la de crear gráficos y salvarlos como tales, comprobar los parámetros que utilizamos en cualquier momento, cuadricular la pantalla, trazar arcos, líneas, círculos y cuadrados, invertir la pantalla, realizar fill o relleno de figuras, posicionar el cursor en cualquier parte de la pantalla y trabajar en modo texto.
Las instrucciones del programa son muy extensas y con ejemplos del manejo.



LEONARDO

THE ARTIST

Un programa con gran velocidad de trabajo y posibilidades bastante amplias.
Entre sus funciones, cabe destacar la de poder utilizarlo como un pequeño procesador de textos, la de crear hasta 7 juegos completos de caracteres visualizando todos ellos en pantalla, la de poder

volcar bloques de 2 * 2 caracteres en pantalla, y aumentar el tamaño de pantalla de la parte que estemos trabajando en ese momento.
La función Fill es una de las más completas que poseen estos programas pudiendo realizar el relleno de figuras con una enorme gama de tramas, que el programa tiene definidas y que nosotros podemos reformar.
En el modo de manejo de gráficos podemos comprobar la animación de cuatro figuras de 2 * 2 caracteres, así como rotar los caracteres, realizar funciones de espejo, invertir el carácter y otras muchas.
Podemos definir el trazo de la línea como lo deseamos y también la forma que tendrá según se va realizando el dibujo.

MELBORNE DRAW

Programa muy útil para la realización de pantallas de fácil manejo.
Entre la multitud de posibilidades cabe destacar la posibilidad de rellenar las dos líneas inferiores de pantalla donde normalmente se visualizan los datos de trabajo, para ello el programa desplaza la posición de la información a la parte superior de la pantalla.
Otras de las opciones de este programa nos permite la realización de caracteres y gráficos.
La pantalla puede ser cuadriculada para utilizarla como guía a la hora de realizar un trabajo.
Si deseamos visualizar una parte del dibujo a mayor tamaño tenemos dos a elegir.
Si deseamos un dibujo en pantalla podemos desplazarla toda en todas las posiciones.
Las demás funciones son similares a los programos existentes de este tipo.


MELBORNE DRAW



VU-3d

VU-3d

Ivestrónica

Si deseamos realizar una figura en tres dimensiones, con este programa podemos realizarlo.
Aporta una figura ejemplo que muestra las posibilidades de trabajo.
Entre ellas podemos hacer mención a aumentar y reducir la figura, rotarla en cuatro direcciones: arriba, abajo, derecha e izquierda.
La posibilidad de coloreado de cada una de las caras visibles es muy amplia, podemos dar color a partir de indicar dónde se encuentra el foco de luz, si es arriba, abajo o en el centro así como derecha, izquierda y centro.
También podemos hacer desaparecer las líneas que se encuentran en la figura por la parte posterior.

Utilidades de duplicación de programas

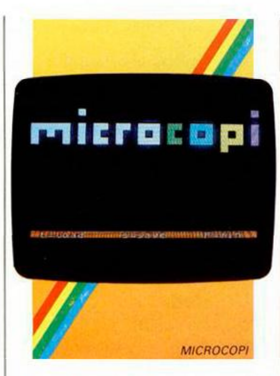
En este cuarto grupo vamos a introducir algunos de los programas más conocidos para la creación de copias de programas, que por su importancia a la hora de trabajar con ellos, sea necesario tener duplicados y no permitan un modo fácil de copia.

MICROCOPI

Microhobby

Con este programa podremos realizar la copia de programas de hasta 41 K en Basic y código máquina.
El programa posee tres opciones que son la carga de un bloque, en el ordenador, copiarlo en cinta y volver al Basic.
Con él no podremos realizar la copia de programas protegidos con turbo o variación de velocidad, así como los programas que no tengan cabecera indicadora.
Se encuentra en el mercado en dos formas: en cinta con los demás programas Microhobby de la revista del número 1 al 4, y en la revista número 1 de MICROHOBBY.

Utilidades



MICROCOPI

TRANS EXPRESS

Babeta, S. A. Co

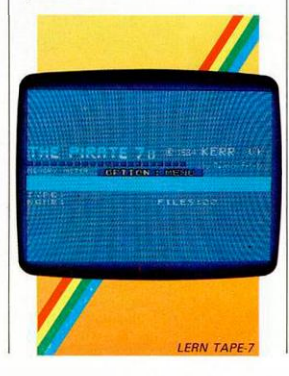
El programa permite cargar más de un programa o bloques de programa en el ordenador.
Puede traspasar programas de cinta a microdrive, de microdrive a microdrive, de microdrive a cinta y de cinta a cinta.
Puede copiar programas sin cabecera que algunos copiones no pueden operar.
Dentro de las utilidades de esta cinta se encuentra la de poder realizar copias que su extensión da superior de 40 K y menor de 48 K.
Se espera la aparición de un programa similar a éste pero con la opción de poder traspasar también a disco y con la posibilidad de copiar programas en alta velocidad y turbo.



TRANS EXPRES

PIRATE 7

Con este copión podemos realizar copias de seguridad de nuestros programas con gran facilidad y permitiendo que el programa a copiar sea bastante extenso.
Se carga en la pantalla y en las franjas centrales presenta el menú de trabajo.
La única pega es la que se nos plantea a la hora de localizar el programa, pues no conocemos posibles distribuidores.



LERN TAPE-7

LERN TAPE 7

El programa Lern es uno de los más completos a la hora de realizar las copias de seguridad poseyendo el menú más extenso de todos ellos.
Podemos cargar cualquier tipo de programa y guardarlo de distintas maneras.
Si deseamos, nos puede servir para traspasar programas que estuviesen en carga rápida y pasarlos a carga normal.
También podemos utilizarlo para la lectura de cabeceras de los programas que deseemos.



PIRATE-7

Utilidades diversas

La última parte de esta guía contiene utilidades diversas que no se pueden incluir en ninguno de los cuatro apartados anteriormente detallados.
Entre estas utilidades se encuentran las que nos pueden ayudar a realizar operaciones matemáticas diversas, test de inteligencia o de conocimiento, programas que nos ayuden a crear música para un programa y otra más que detallaremos a continuación.

O SON SEIKOSHA...


CARNAVAL SITGES


OPERA CHINA


DIOS NEPALI


SEIKOSHA GP


OPERA CHINA


POPULAR HONG KONG


SEIKOSHA SP


POPULAR CHINA


CARNAVAL RIO


SEIKOSHA MP


CERAMICA MANISES


OPERA JAPONESA


SEIKOSHA BP


SATIRO


POPULAR JAPON

...O SON MASCARAS

GP-50 •	La pequeña 40 cps. Papel normal con interface paralelo, serial y Spectrum	17.990 ptas.
GP-700 *	La de color 50 cps. 7 colores. 80 columnas. Tracción y fricción. Papel de 10 pulgadas	64.990 ptas.
SP-1.000 *	La programable 100 cps.24 cps en alta calidad 96 cart. programables en RAM. Introductor hoja a hoja.♦	64.990 ptas.
SP-1.000AS	La programable 100 cps.24 cps en alta calidad con interface RS-232. Introductor hoja a hoja.♦	59.900 ptas.
MP-1.300AI	La polivalente 300 cps, 60 cps en alta calidad, interface paralelo y RS-232. Introductor hoja a hoja.♦&	119.900 ptas.
BP-5.200 *	La de oficina 200 cps, 106 en alta calidad.Buffer 4K.Carro de 15".Tracción y fricción.♦	199.900 ptas.
BP-5.420 *	La más rápida 420 cps. 106 cps en alta calidad. Buffer de 18K. Paralelo y RS-232.♦	299.900 ptas.

Interfaces: Serie RS-232C, Spectrum, IBM, COMMODORE, MSX, QL, Apple Macintosh, HP-IB
♦ Disponen de introductor automático de documentos opcional.
& Dispone de Kit opcional de color

* con interface paralelo
• con interface Spectrum

Nota: I.V.A. 12%, no incluido en los precios arriba indicados

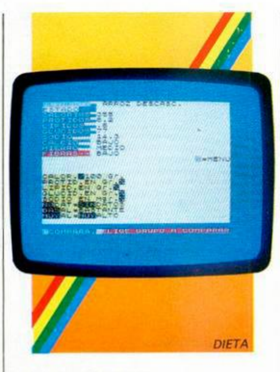
Avda. Blasco Ibáñez, 116
Tel. (96) 372. 88. 89
Telex 62220 - 46022 VALENCIA

Muntaner, 60-2.º-4.ª
Tel. (93) 323. 32. 19
08011 BARCELONA

Agustin de Foxá, 25-3.º-A
Tels. (91) 733. 57. 00 - 733. 56. 50
28036 MADRID



Utilidades


DIETA

DIETA Y SALUD

Boalox

La cinta contiene dos programas, uno en cada cara, que nos serán de ayuda a la hora de comprobar el estado físico de nuestro cuerpo. Con el programa Dieta podremos conocer la tabla de nutrición de los alimentos que deseemos, dicha tabla está basada en la de la Casa Alter.



SALUD

Podremos también saber cómo es la forma más usual de consumo de un alimento englobados en distintos grupos, cómo son los vegetales, las carnes, los pescados o las bebidas.
Otra de las operaciones que realiza este programa es la de comparar entre sí dos alimentos.
El programa Salud sirve sobre todo para realizar un pequeño chequeo de nuestro cuerpo. Tenemos la posibilidad de comprobar cómo estamos en relación altura y peso, la audiometría, la ostetricia y los parámetros de pediatría.

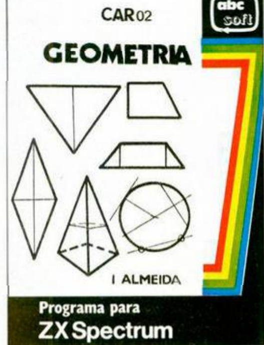
ESTADISTICA

Boalox

Es una utilidad gráfica y matemática para el tratamiento de datos estadísticos.
El paquete consta de dos cintas

ESTADISTICA

con dos programas de utilidades distintas.
La utilización es de fácil manejo y en los programas se acompañan programas de ejemplo y manual de utilidad.
Entre las operaciones que podemos realizar se encuentran: cálculo de la media, desviación típica, valor máximo y mínimo de cada variable o los valores de la recta de regresión.
Muy útil en actividades comerciales en la que se manejen datos estadísticos.



GEOMETRIA

GEOMETRIA

ABC Soft

Con este programa podremos enseñar a nuestros hijos los conocimientos más fundamentales de la geometría.
El programa, además de esa utilidad, permite la realización de operaciones geométricas de todo tipo, pudiendo realizar cálculos como el de áreas y volúmenes.

MELODIAN

Music Soft

Programa de una casa dedicada exclusivamente a los programas de utilidad musical.
Entre los que se encuentran a

disposición podremos hallar programas para tocar música con el Spectrum y para leer las notas según se van introduendo en el ordenador.
Otro de los programas de esta



MELODIAN

casa es el MINI COMPOSITOR con el que podemos crear y grabar música pudiendo retocar la partitura de las notas, cambiar el tempo y otras opciones útiles a la hora de crear música.

Si deseamos hacer un fichero de partituras o discos, podemos utilizar los programas de fichero DISCOGRAFICO y MUSICAL de la misma casa, con la que lograremos ordenar nuestra colección de discos que según va creciendo va siendo más difícil localizar, o saber si lo poseemos.

OPTOCHECK

Boalox

Con este programa podremos realizar chequeos del estado de los parámetros de la visión.
Contiene instrucciones para el manejo de todas sus utilidades como son la agudeza visual de adultos y de niños, radios de astigmatismo, poder de convergencia, punto próximo de acomodación, visión binocular y fusión, pampimetria automática y manuãl.
El programa puede ser de utilidad para médicos en general, ópticos, psicotécnicos...

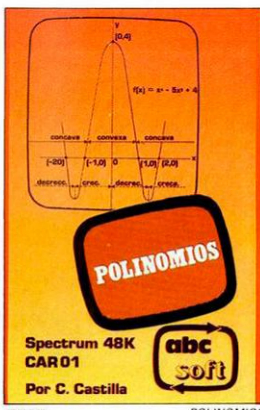


OPTOCHECK

POLINOMIOS

ABC Soft

Ideal para la utilización didáctica, sobre todo para demostrar prácticamente el manejo de los polinomios.
El programa consta de dos menús de trabajo, entre ellos: calcular una integral, raíces enteras, acotar raíces a mínimos y máximos, hallar la derivada enésima, crear un polinomio, etc.
Muy aconsejable para la utilización en grupos, tipo escuela y muy útil para realizar las complicadas operaciones de polígonos y derivadas.



POLINOMIOS

TEST

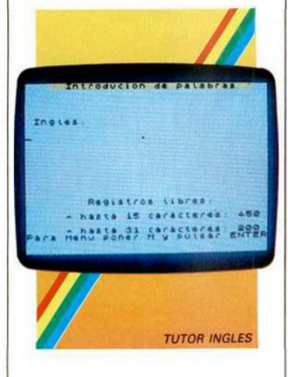
Boalox

La cinta con el nombre de Test consta de varios programas que como su nombre indica, realizan test de distintas materias.
La utilidad viene dada porque con estos test podemos saber cómo



TEXT

estamos de conocimientos generales sobre distintas materias desde la historia hasta el inglés y el francés.
El número de preguntas y respuestas se puede fácilmente transformar a nuestro antojo.
En el primero de los test sólo tenemos que contestar verdadero o falso, en el segundo y tercer test tenemos que acertar la respuesta verdadera y los dos últimos test son para la práctica del vocabulario inglés y francés.



TUTOR INGLES

TUTOR DE INGLES

ABC Soft

Programa Monitor de inglés que nos permite comprobar, mejorar e incluso aprender el inglés.
El programa funciona de una manera similar a un archivo de palabras en español-inglés.
Entre las posibilidades del programa está la de poder crear ficheros con distintos datas, pudiendo incrementar el fichero o incluir palabras que nos interesa repasar.
En la zona de escritura podemos introducir la n pulsando el 9 y obteniendo así la traducción correcta de las palabras al español.
En el menú nos aparecen un total de ocho opciones entre las que se encuentran las de grabar y cargar un fichero, crear un nuevo fichero, practicar, introducción de palabras, diccionario y listado del fichero.

VOCABULARIO ALEMAN

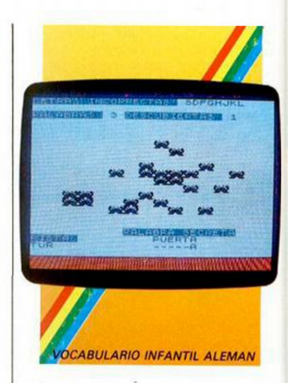
Investrónica

El programa es una guía práctica para familiarizarte con el idioma alemán.
Incluye un total de 3.600 palabras, la mitad en alemán y la otra mitad en castellano. Si deseamos añadir alguna palabra que consideremos importante, podremos introducir un total de hasta 200 palabras más.
Según vamos traduciendo la palabra contestada ayudamos a poner en órbita un cohete espacial, con lo que proporciona a los niños, además de un apartado de trabajo, un incentivo de distracción.

VOCABULARIO INFANTIL ALEMAN

Investrónica

Para empezar a manejarse en el idioma alemán nos será de gran utilidad este programa.
Puede ser utilizado de tres maneras, que se pueden tomar como fases de aprendizaje. En la primera, debemos descubrir cuál es la traducción de una palabra castellana al alemán, en la segunda opción deberemos encontrar el significado de una palabra alemana en castellano y



VOCABULARIO INFANTIL ALEMAN

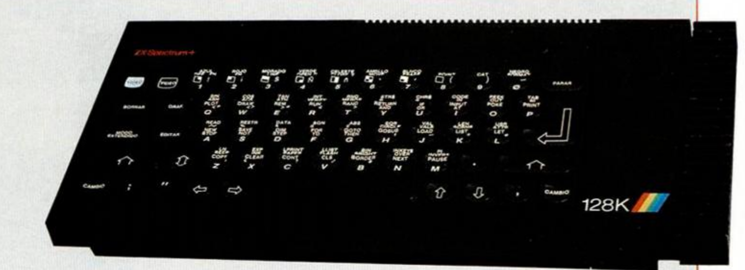
por último, el ordenador pregunta por palabras aleatorias en ambos idiomas.
Si tenemos la necesidad de usar algún carácter especial del alemán, tendremos unas teclas definidas para su utilización.

VOCABULARIO EN INGLES

Investrónica

El programa en su funcionamiento nos recuerda mucho al juego del ahorcado.
En la pantalla aparece una bomba y nosotros debemos descubrir la traducción de la palabra de manera que la mano con el mechero no encienda la bomba y nos explote.
La forma de manejo es similar a la del vocabulario infantil en alemán.

Direcciones de las casas productoras



ABC SOFT
C/. Santa Cruz de Marcenado, número 31, 3, 14
Tel. 248 82 13
28015 Madrid

INVESTRONICA
Tomás Bretón, 60
Tel. 468 03 00
Madrid

HISOFT
180 Higth Street North
Dunstable Beds, LU6 IAT
Tel. (0582) 69 64 21
Inglaterra 2

BABETA, S. A. Co
Galileo, 25
Tel. 447 97 51 - 447 98 09
28015 Madrid

VENTAMATIC
Avda. de Rhode, 168
Rosas (Gerona)

BOALOX
General Franco, 87, 2
Tel. (988) 22 16 47
32003 Orense

MUSIC SOFT
Magallanes, 27
28015 Madrid

TASMAN
Springfield House, Hyde Terrace
Leeds LS2 9LN
Tel. (0532) 43 83 01
Inglaterra

MICROBYTE
P. Castellana, 179, 1
Tel. 442 54 33
28046 Madrid

MICROGESA
C/. Silva, 5, 4
Tel. 242 24 71
28013 Madrid

SICLAIR STORE
C/. Bravo Murilla, 2
Tel. 446 62 31
Madrid

Por último, recordamos que entre la guía de estas utilidades no están todas las que son, pero sí las más conocidas.

Transformaciones de planos

Alejandro JULVEZ
Marcos ORTIZ



Mover, ampliar, girar y reproducir un objeto en la pantalla es una tarea difícil. Con la apliación de las matemáticas, como veréis el problema queda reducido a una simple codificación.

En primer lugar debemos dar un ligero repaso a la idea y concepto de lo que representa una transformación en el plano.

Las transformaciones de tipo lineal, que son las que nosotros vamos a utilizar son tres (traslación, giro y homotecias). La aplicación de dichas transformaciones nos va a permitir jugar de una manera eficaz y cómoda con los objetos definidos por nosotros mismos.

TRASLACION: supongamos un vector libre del plano A, mediante el vector A definimos una transformación, llamada traslación, de forma que a cada punto X del plano le haremos corresponder un X' tal que el vector XX' = A. Las ecuaciones de la traslación son
x' = x + a
y' = y + b siendo (a, b) las coordenadas del vector A y (x, y), (x', y') las coordenadas de los puntos X y X' respectivamente.

A cada punto X le corresponde un único X' ya que existe un único representante del vector A que tiene origen en X. Al punto X' se le llama transformado de dicha traslación.

GIROS: dado un punto 0 del plano y un ángulo alpha se llama giro a la transformación del plano que hace corresponder a cada punto X un punto X' girado un ángulo alpha con respecto al origen de coordenadas. Las ecuaciones del giro son:
x' = x cos alpha—y sen alpha
y' = x sen alpha + y cos alpha

HOMOTECIAS: dado un punto 0 del plano y un número k distinto de cero, se llama homotecia de centro 0 y razón k a la transformación del plano que cambia cada punto X en otro X' tal que el vector OX' = K * OX.

Una vez metidos en materia, diremos que la idea principal es asimilar los pixels del espacio gráfico del spectrum con los puntos del plano euclídeo. Para nosotros el plano euclídeo tendrá 256 * 176 puntos, que corresponden al número de pixels del spectrum. Situaremos el origen de coordenadas en el punto (128, 88) de forma que la pantalla nos quede dividida en 4 cuadrantes. Echemos una ojeada al eje x, desde el pixels Ø hasta el 128 representa para nosotros el semieje negativo de las x, desde el 128 hasta 255 el positivo. Igualmente ocurre con el eje y.

Lógicamente y con el fin de aprovechar al máximo las posibilidades del marco gráfico, el eje y tiene menos puntos que el x.

Ahora se nos presenta el problema de la representación de un objeto, para ello necesitaremos dos vectores, X e Y, con el fin de guardar las coordenadas de los vértices del objeto, y una matriz que nos llevará un control de las líneas de unión entre los distintos vértices. La definición del objeto la haremos con referencia al punto (Ø, Ø) encargándose una subrutina, concretamente la 12ØØ, del paso a las coordenadas de nuestro sistema de referencia con origen (128, 88).

Para la creación o definición del objeto utilizamos la subrutina 1ØØØ.

La subrutina 1ØØØ necesita una serie de parámetros:

nv → guarda el número de vértices del objeto, servirá para dimensionar los vectores X e Y.

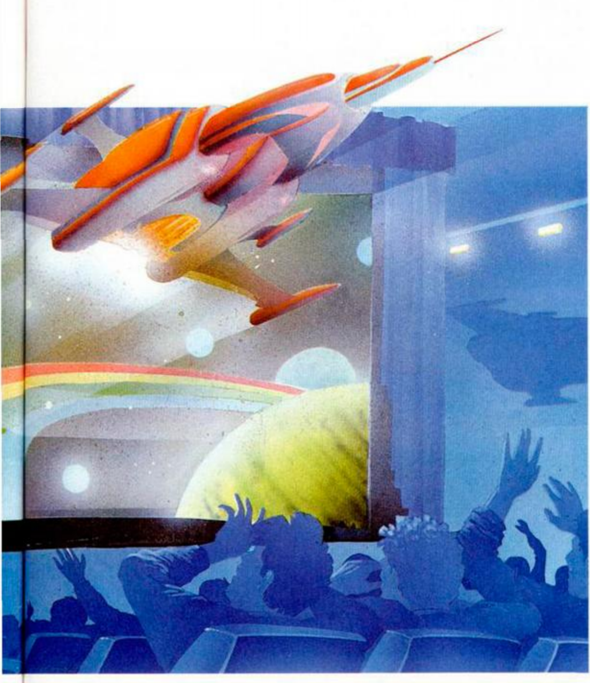
nl → guarda el número de líneas de unión y sirve para dimensionar la matriz de líneas Z.

A$ → indica si se desean o no líneas de unión.

xorigen, yorigen → reflejan las coordenadas del origen (128, 88).

LOAD" CODE 32.000

En el caso de los vértices es necesario la introducción de todas las coordenadas, pero en el caso de las líneas y con el fin de facilitar el proceso de diseño, basta con introducir un Ø como último dato para indicar de esta forma que no hay más líneas aunque la variable nl sea más grande. Los vértices empiezan a numerarse a partir del 1.



Veamos un ejemplo: para definir un cuadrado situado en pantalla con las coordenadas de sus vértices (5, 26) (5, 5Ø) (3Ø, 5Ø) (3Ø, 26) introducimos las coordenadas x e y de cada vértice y después las líneas de unión en la forma: unir vértice número 1 con el vértice número 2, el 2 con el 3, el 3 con el 4 y el 4 con el 1.

Finalmente, teclear esta línea para ver su aparición en pantalla:

```
1Ø GO SUB 1ØØØ:GO SUB 12ØØ:GO SUB 13ØØ:STOP
```

Pasemos ahora a comentar cada una de las siguientes subrutinas.

SUBRUTINA 1ØØØ:

Se emplea para la definición del objeto, puede omitirse, pero en ese caso debemos tener en cuenta las variables anteriormente mencionadas e inicializarlas con sus valores correspondientes para que puedan trabajar las demás subrutinas.

Esta subrutina se omite cuando el objeto es muy complicado y requiere un gran número de datos, con el fin de no teclearlos todos los introducimos en líneas data. Más adelante vemos un ejemplo de utilización.

SUBRUTINA 12ØØ:

La misión de esta subrutina es el paso a nuestro sistema coordenado. Aunque para la impresión las coordenadas son absolutas y no relativas a nuestro origen, es necesario esta conversión con el fin de que puedan trabajar las distintas subrutinas.

SUBRUTINA 13ØØ:

Imprime el objeto junto con sus líneas en caso de que se haya seleccionado esta opción.

SUBRUTINA 14ØØ:

Esta subrutina trata la primera de las transformaciones antes comentadas, que es la traslación.

A esta subrutina le debemos pasar como parámetros las coordenadas de un vector que efectúa la traslación. Las coordenadas de este vector se pasan a la subrutina en las variables XA, YA. Es importante decir que las coordenadas de dicho vector vienen referidas al origen en el punto (128, 88), es decir, pueden tomar los valores:

(—128 < = XA < = 127)
(—88 < = YA < = 87)

SUBRUTINA 15ØØ:

Esta subrutina se encarga de la rotación o giro. Para ello, debemos pasar a la subrutina un parámetro que es el ángulo de giro con respecto al origen, situado en el punto (128, 88).

La variable alpha es la encargada de almacenar el valor del ángulo en radianes (Ø < = alpha < = 2*PI).

SUBRUTINA 16ØØ:

Se encarga del dibujo de los ejes coordenados. Su utilización es opcional y su empleo es meramente aclarativo de qué es lo que ocurre en pantalla, con una referencia al origen.

SUBRUTINA 17ØØ:

Se encarga de las homotecias en el plano. Se le pasan 2 parámetros FESCX y FESCY donde registramos el valor a multiplicar las coordenadas de los vértices. No tienen por qué ser iguales ambos valores tratándose en este caso de un factor de escala más que de una homotecia.

La homotecia es una biyección del plano y el único punto invariante por la homotecia será el origen. Sólo en el caso f FESCX = FESCY = 1 serán todos los puntos invariantes.

Una homotecia transforma una recta en otra paralela a ella, luego transforma un ángulo en otro ángulo igual por tener sus lados paralelos a los del lado. Transforma un segmento en otro proporcional al anterior con una proporcionalidad de ABS(K) siendo K la razón.

SUBRUTINA 18ØØ:

Se encarga de la transformación y giro de un objeto con respecto a un punto de mira del observador con un ángulo dado beta.

Los parámetros que se le pasan a esta subrutina son el punto de mira, pero referido al origen en el punto (Ø, Ø) y por tanto, los posibles valores que pueden tomar son:

MX →(Ø < = MX < = 255)
MY →(Ø < = MY < = 175)

También se le pasa el ángulo beta (Ø < = beta < = 2*PI).

SUBRUTINA 19ØØ:

Proporciona una simetría del objeto con respecto al eje X.

SUBRUTINA 194Ø:

Proporciona una simetría del objeto con respecto al eje Y.

La simetría con respecto al origen de coordenadas se consigue girando el objeto con respecto a dicho origen un ángulo alpha = PI es decir, utilizando la subrutina 15ØØ con alpha = PI.

Una vez explicadas las subrutinas necesarias, vamos a pasar a ver algunos ejemplos de utilización.

TRASLACIONES

Ejemplo 1: trasladar un punto (8, 19) mediante el vector (14, 2).

5 REM IMPRESION EJES:GO SUB 16ØØ

1Ø REM DEFINICION DEL PUNTO:GO SUB 1ØØØ

2Ø REM PASO A NUESTRO SISTEMA DE REFERENCIA:GO SUB 12ØØ

3Ø REM REM IMPRESION DEL PUNTO:HO SUB 13ØØ

4Ø REM TRASLACION:LE XA = =14:LET YA=2:GO SUB 14ØØ:GO SUB 13ØØ

5Ø STOP

En este ejemplo, a la pregunta: ¿desea líneas? respondemos NO.

Ejemplo 2: trasladar el cuadro de vértices (5, 26) (5, 5Ø) (3Ø, 5Ø) (3Ø, 26) mediante el vector (12Ø, 8Ø).

1Ø GO SUB 16ØØ

2Ø GO SUB 1ØØØ:GO SUB 12ØØ:GO SUB 13ØØ

3Ø LET XA = 12Ø:LET YA = 8Ø:GO SUB 14ØØ:GO SUB 13ØØ:STOP

En este caso sí necesitamos la existencia de líneas para unir los vértices del cuadrado. Uniremos los vértices:

1 con 2
2 con 3
3 con 4
4 con 1

ROTACION

Ejemplo 1: definimos una flecha de vértices:

(151, 9Ø) (17Ø, 9Ø) (166, 93) (166, 87) deseamos girar con respecto al origen dicha flecha un ángulo de 9Ø grados.

1Ø GO SUB 16ØØ

2Ø GO SUB 1ØØØ:GO SUB 12ØØ:GO SUB 13ØØ

LISTADO 1

```
1000 REM **********************
1005 REM DEFINICION
1010 INPUT "numero de vertices:"
;nv
1015 INPUT "DESEA LINEAS?";A$
1016 IF A$="NO" THEN GO TO 1030
1020 INPUT "numero de lineas:";n
l
1025 DIM Z(2,NL)
1030 DIM x(nv): DIM y(nv)
1040 FOR i=1 TO nv
1045 PRINT AT 21,10;"VERTICE ";I
1050 INPUT "coordenada(x)";x(i)
1060 IF x(i)>255 OR x(i)<0 THEN
GO TO 1050
1070 INPUT "coordenada(y)";y(i)
1080 IF y(i)>175 OR y(i)<0 THEN
GO TO 1070
1090 NEXT i: PRINT AT 21,10;"
 "
1095 IF A$="NO" THEN GO TO 1160
1100 FOR i=1 TO nl
1110 INPUT "unir vertice numero:
";z(1,i)
1115 IF Z(1,I)=0 THEN GO TO 1160
1120 IF z(1,i)>nv OR z(1,i)<1 TH
EN GO TO 1110
1130 INPUT "con vertice numero:"
;z(2,i)
1140 IF z(2,i)>nv OR z(2,i)<1 TH
EN GO TO 1130
1150 NEXT i
1160 RETURN
1200 REM paso a nuestro sistema
de referencia
1210 FOR i=1 TO nv
1220 LET x(i)=x(i)-xorigen
1230 LET y(i)=y(i)-yorigen
1240 NEXT i
1245 RETURN
1300 REM imprime objeto: OVER 1
1310 FOR i=1 TO nv
1320 PLOT xorigen+x(i),yorigen+y
(i)
1330 NEXT i
1345 IF A$="NO" THEN GO TO 1380
1350 FOR i=1 TO nl
1355 IF Z(1,I)=0 THEN GO TO 1380
1357 PLOT X(Z(1,I))+XORIGEN,Y(Z(
1,I))+YORIGEN
1360 DRAW x(z(2,i))-x(z(1,i)),y(
z(2,i))-y(z(1,i))
1370 NEXT i
1380 OVER 0: RETURN
1400 REM traslacion
1410 FOR i=1 TO nv
1420 LET x(i)=INT (x(i)+xa)
1430 IF x(i)>127 THEN LET x(i)=1
27
1440 IF x(i)<-128 THEN LET x(i)=
-128
1450 LET y(i)=INT (y(i)+ya)
1460 IF y(i)>87 THEN LET y(i)=87
1470 IF y(i)<-88 THEN LET y(i)=-
88
1480 NEXT i
1490 RETURN
1500 REM rotacion
1510 FOR i=1 TO nv
1515 LET AUX=X(I)
1520 LET x(i)=INT ((x(i)*COS alp
ha-y(i)*SIN alpha)+0.5)
1530 IF x(i)>127 THEN LET x(i)=1
27
1540 IF x(i)<-128 THEN LET x(i)=
-128
1550 LET y(i)=INT ((AUX*SIN alph
a+y(i)*COS alpha)+0.5)
1560 IF y(i)>87 THEN LET y(i)=87
1570 IF y(i)<-88 THEN LET y(i)=-
88
1580 NEXT i
1590 RETURN
1600 REM dibuja ejes
1610 FOR i=0 TO 255 STEP 3
1620 PLOT i,88: NEXT i
1630 FOR i=0 TO 175 STEP 2
1640 PLOT 128,i: NEXT i
1645 PRINT AT 0,15;"Y": PRINT AT
 11,31;"X"
1650 RETURN
1700 REM homotecias
1710 FOR I=1 TO NV
1720 LET X(I)=INT (X(I)*FESCX)
1730 IF X(I)>127 THEN LET X(I)=1
27
1740 IF X(I)<-128 THEN LET X(I)=
-128
1750 LET Y(I)=INT (Y(I)*FESCY)
1760 IF Y(I)>87 THEN LET Y(I)=87
1770 IF Y(I)<-88 THEN LET Y(I)=-
88
1780 NEXT I
1790 RETURN
1800 REM PUNTO DE MIRA
1810 LET MX=MX-XORIGEN: LET MY=M
Y-YORIGEN
1820 LET XA=-MX: LET YA=-MY
1830 GO SUB 1400
1840 LET ALPHA=-BETA: GO SUB 150
0
1850 RETURN
1900 REM SIMETRIA EJE X
1910 FOR I=1 TO NV
1920 LET Y(I)=-Y(I): NEXT I
1930 RETURN
1940 REM SIMETRIA EJE Y
1950 FOR I=1 TO NV
1960 LET X(I)=-X(I): NEXT I
1970 RETURN
1980 REM **********************
```



```
3Ø LET ALPHA=PI/2:GO SUB
15ØØ:GO SUB 13ØØ
4Ø STOP
```

Debiendo unirse los vértices:

1 con 2
2 con 3
2 con 4.

En este caso es necesario la utilización de líneas de unión.

El efecto es el reflejado en la figura R1.

Ejemplo 2: vamos a dibujar un pirulí parecido al de televisión, los vértices son los siguientes:

(75, 14Ø) (75, 13Ø) (8Ø, 12Ø) (9Ø, 116) (86, 11Ø) (81, 1Ø6) (81, 71) (9Ø, 71) (9Ø, 68) (61, 68) (61, 71) (7Ø, 71) (7Ø, 1Ø6) (65, 11Ø) (61, 116) (7Ø, 12Ø)

las uniones que se deben realizar son:

1-2-3-4-5-6-7-8-9-1Ø-11-12-13-14-15-16-1

A dicho pirulí le vamos a aplicar una traslación con un vector (4Ø, 2Ø), después otra traslación de vector (4Ø, Ø) y a continuación, un giro de ángulo alpha = PI.

```
1Ø GO SUB 16ØØ
2Ø GO SUB 1ØØØ:GO SUB 12ØØ:GO
SUB 13ØØ
3Ø LET XA=4Ø:LET YA=2Ø:GO
SUB 14ØØ:GO SUB 13ØØ
4Ø LET XA=4Ø:LET YA=Ø:GO
SUB 14ØØ:GO SUB 13ØØ
5Ø LET ALPHA=PI:GO SUB
15ØØ:GO SUB 13ØØ
6Ø STOP
```

Figura R2.

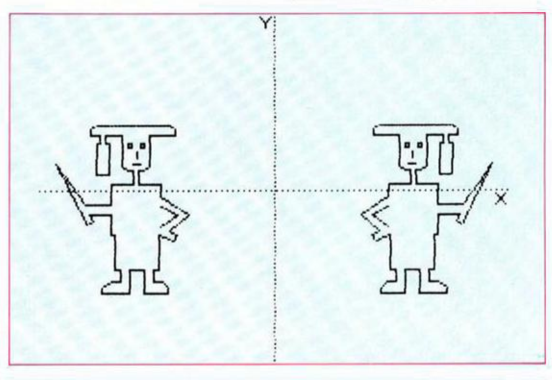
Como habréis podido observar, la definición de un objeto es una tarea pesada y con un margen de equivocación alto. Os aconsejamos la utilización de papel milimetrado y en el caso de un objeto complicado, la utilización como antes mencionamos de una rutina de carga del objeto más cómoda. En el ejemplo siguiente vamos a definir un marciano de unas dimensiones considerables y veréis el trabajo que lleva su definición. En cualquier caso. el trabajo por primera vez para la creación es inevitable, pero con estas subrutinas nos ahorramos tener que calcular a mano las coordenadas cuando movamos el objeto, lo giremos o ampliemos. Este grupo de subrutinas nos permiten el trabajar de forma eficaz con objetos definidos por nosotros mismos y que luego podemos utilizar en nuestros programas.

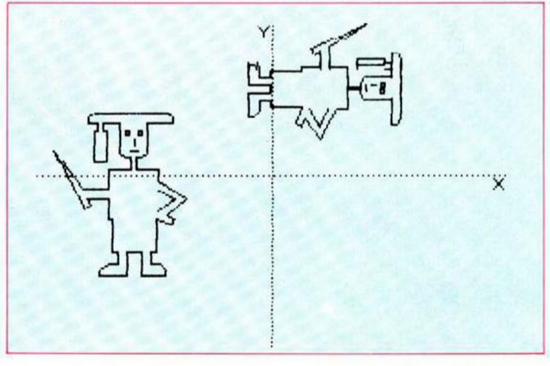
Adelante con el marciano:

Programa R3.

Figura R3.

Realizamos algún experimento con nuestro visitante para que no le queden ganas de volver a vernos.





LISTADO 2

```
10 REM definicion del marciano
20 LET xorigen=128: LET yorige
n=88: LET nv=72: LET nl=90: LET
a$="SI"
30 DIM x(nv): DIM y(nv): DIM z
(2,nl)
35 REM vertices
40 DATA 31,120,71,120,74,118,7
4,115,60,115,60,100,57,98,54,98,
54,91,66,91,66,85,81,76,71,68,75
,66,73,63,65,66,65,49
50 DATA 62,49,62,43,67,43,70,4
0,70,37,57,37,57,49,49,49,49,37,
34,37,34,40,39,43,44,43,44,49,41
,49
60 DATA 41,66,39,66,39,76,27,7
6,29,73,26,71,9,101,25,81,39,81,
39,91,51,91,51,98,48,98,45,100,4
5,115,36,115,36,111
70 DATA 38,111,38,96,31,96,31,
111,33,111,33,115,28,115,28,118,
48,111,50,111,50,109,48,109
80 DATA 55,111,57,111,57,109,5
5,109,53,107,53,104,51,101,55,10
1,66,80,75,76,66,70
85 REM lineas
90 DATA 1,2,2,3,3,4,4,5,5,6,6,
7,7,8,8,9,9,10,10,11,11,12,12,13
,13,14,14,15,15,16,16,17,17,18,1
8,19,19,20,20,21,21,22,22,23,23,
24,24,25,25,26,26,27,27,28,28,29
,29,30,30,31,31,32,32,33,33,34,3
4,35,35,36,36,37,37,38,38,39,39,
40
100 DATA 40,41,41,42,42,43,43,4
4,44,45,45,46,46,47,47,48,48,49,
49,50,50,51,51,52,52,53,53,54,54
,55,55,56,56,57,57,1,58,59,59,60
,60,61,61,58,62,63,63,64,64,65,6
5,62,66,67,68,69,70,71,71,72,0
110 FOR i=1 TO nv
120 READ X(I): READ Y(I)
125 NEXT I
130 FOR i=1 TO nl
140 READ z(1,i): IF z(1,i)=0 TH
EN GO TO 160
145 READ z(2,i)
150 NEXT i
160 GO SUB 1600: GO SUB 1200: G
O SUB 1300: STOP
1000 REM ***********************
1005 REM DEFINICION
1010 INPUT "numero de vertices:"
;nv
1015 INPUT "DESEA LINEAS?";A$
1016 IF A$="NO" THEN GO TO 1030
1020 INPUT "numero de lineas:";n
l
1025 DIM Z(2,NL)
1030 DIM x(nv): DIM y(nv)
1040 FOR i=1 TO nv
1045 PRINT AT 21,10;"VERTICE ";I
1050 INPUT "coordenada(x)";X(i)
1060 IF x(i)>255 OR x(i)<0 THEN
GO TO 1050
1070 INPUT "coordenada(y)";y(i)
1080 IF y(i)>175 OR y(i)<0 THEN
GO TO 1070
1090 NEXT i: PRINT AT 21,10;"
        "
1095 IF A$="NO" THEN GO TO 1160
1100 FOR i=1 TO nl
1110 INPUT "unir vertice numero:
";z(1,i)
1115 IF Z(1,I)=0 THEN GO TO 1160
1120 IF z(1,i)>nv OR z(1,i)<1 TH
EN GO TO 1110
1130 INPUT "con vertice numero:"
;z(2,i)
1140 IF z(2,i)>nv OR z(2,i)<1 TH
EN GO TO 1130
1150 NEXT i
1160 RETURN
1200 REM paso a nuestro sistema
de referencia
1210 FOR i=1 TO nv
1220 LET x(i)=x(i)-xorigen
1230 LET y(i)=y(i)-yorigen
1240 NEXT i
1245 RETURN
1300 REM imprime objeto: OVER 1
1310 FOR i=1 TO nv
1320 PLOT xorigen+x(i),yorigen+y
(i)
1330 NEXT i
1345 IF A$="NO" THEN GO TO 1380
1350 FOR i=1 TO nl
1355 IF Z(1,I)=0 THEN GO TO 1380
1357 PLOT X(Z(1,I))+XORIGEN,Y(Z(
1,I))+YORIGEN
1360 DRAW x(z(2,i))-x(z(1,i)),y(
z(2,i))-y(z(1,i))
1370 NEXT i
1380 OVER 0: RETURN
1400 REM traslacion
1410 FOR i=1 TO nv
1420 LET x(i)=INT (x(i)+xa)
1430 IF x(i)>127 THEN LET x(i)=1
27
1440 IF x(i)<-128 THEN LET x(i)=
-128
1450 LET y(i)=INT (y(i)+ya)
1460 IF y(i)>87 THEN LET y(i)=87
1470 IF y(i)<-88 THEN LET y(i)=-
88
1480 NEXT i
1490 RETURN
1500 REM rotacion
1510 FOR i=1 TO nv
1515 LET AUX=X(I)
1520 LET x(i)=INT ((x(i)*COS alp
ha-y(i)*SIN alpha)+0.5)
1530 IF x(i)>127 THEN LET x(i)=1
27
1540 IF x(i)<-128 THEN LET x(i)=
-128
1550 LET y(i)=INT ((AUX*SIN alph
a+y(i)*COS alpha)+0.5)
1560 IF y(i)>87 THEN LET y(i)=87
1570 IF y(i)<-88 THEN LET y(i)=-
88
1580 NEXT i
1590 RETURN
1600 REM dibuja ejes
1610 FOR i=0 TO 255 STEP 3
1620 PLOT i,88: NEXT i
1630 FOR i=0 TO 175 STEP 2
1640 PLOT 128,i: NEXT i
1645 PRINT AT 0,15;"Y": PRINT AT
 11,31;"X"
1650 RETURN
1700 REM homotecias
1710 FOR I=1 TO NV
1720 LET X(I)=INT (X(I)*FESCX)
1730 IF X(I)>127 THEN LET X(I)=1
27
1740 IF X(I)<-128 THEN LET X(I)=
-128
1750 LET Y(I)=INT (Y(I)*FESCY)
1760 IF Y(I)>87 THEN LET Y(I)=87
1770 IF Y(I)<-88 THEN LET Y(I)=-
88
1780 NEXT I
1790 RETURN
1800 REM PUNTO DE MIRA
1810 LET MX=MX-XORIGEN: LET MY=M
Y-YORIGEN
1820 LET XA=-MX: LET YA=-MY
1830 GO SUB 1400
1840 LET ALPHA=-BETA: GO SUB 150
0
1850 RETURN
1900 REM SIMETRIA EJE X
1910 FOR I=1 TO NV
1920 LET Y(I)=-Y(I): NEXT I
1930 RETURN
1940 REM SIMETRIA EJE Y
1950 FOR I=1 TO NV
1960 LET X(I)=-X(I): NEXT I
1970 RETURN
1980 REM ***********************
```

Vamos a moverlo con un vector (12Ø, 6Ø) a ver qué ocurre. Introducir:

17Ø LET XA = 12Ø:LET YA = 6Ø:GO SUB 14ØØ:GO SUB 13ØØ

18Ø STOP

Observamos que el gorro de nuestro amigo ha quedado chafado, es lógico ya que la traslación pretendía ir más allá de los límites permitidos.

Echar un vistazo a la rutina 14ØØ y veréis qué es lo que ha ocurrido.

Parece ser que nuestro amigo no ha quedado muy impresionado por el movimiento a que le hemos sometido así que vamos a aprovecharnos de nuestras propiedades mágicas y lo vamos a reducir de tamaño.

17Ø LET FESCX = Ø.5:LET FESCY = Ø.5:GO SUB 17ØØ:GO SUB 13ØØ

18Ø STOP

Figura R4.

También lo podemos ampliar de tamaño, pero antes lo desplazaremos mediante una traslación de vector (6Ø, Ø)

17Ø LET XA = 6Ø:LET YA = Ø:HO SUB 14ØØ

18Ø CLS

19Ø LET FESCX = 2:LFESCY = 2: GO SUB 17ØØ

2ØØ GO SUB 13ØØ

21Ø STOP

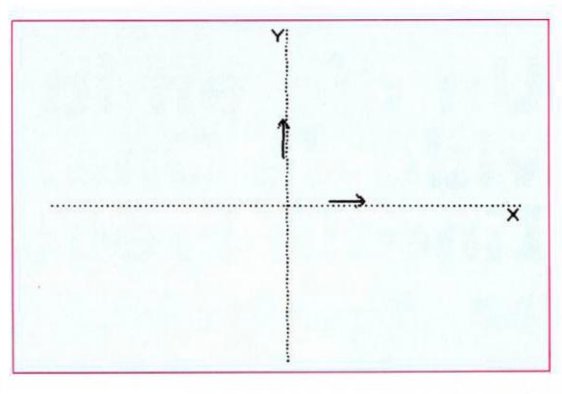
Parece que nuestro amigo está empezando a sentirse mal, pero todavía le queda una dura estancia.

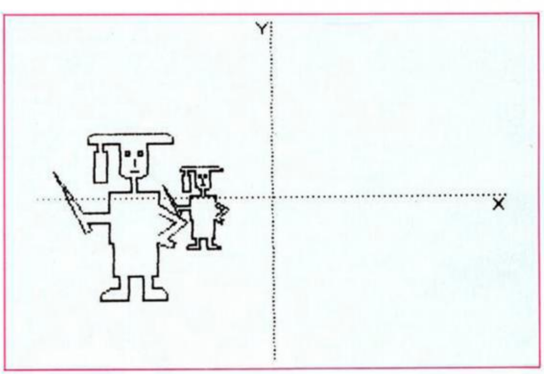
Vamos a ponerlo en nuestro punto de mira sobre el punto (1ØØ, 5Ø) y con un ángulo de 9Ø grados vamos a observarlo:

17Ø LET MX = 1ØØ:LET MY = 5Ø:LET BETA = PI/2:GO SUB 18ØØ

18Ø GO SUB 13ØØ

19Ø STOP





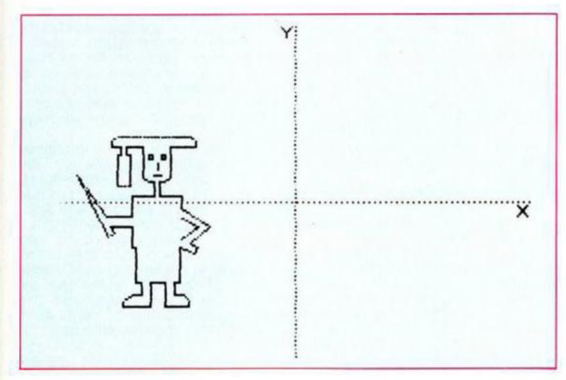


Figura R5.

Por último, vamos a darle la impresión a nuestro amigo de que aparece duplicado, realizando una simetría con respecto al eje Y.

17Ø GO SUB 194Ø:GO SUB 13ØØ
18Ø STOP

Figura R6.

Una buena aportación vuestra a este conjunto de rutinas sería el poder dar color a los objetos y el rellenarlos.

También podéis utilizar estas rutinas para el movimiento animado por medio de la aparición de pantallas grabadas con estas subrutinas.

Para aquellos de vosotros que os animéis, intentar crear un juego de subrutinas similar, pero de las transformaciones del espacio euclídeo. Animo y buena suerte.

Un día en la vida de Onerr Lincoln Freud

José María Díaz

Despertó despacio, como a desgana. Desde algún lejano y oculto rincón de su mente, los musicales apremios de su programa simbiótico le arrancaron, suave pero implacablemente, de las garras de algodón del sueño.

Podía sentir cómo iba tomando conciencia de su cuerpo, músculo a músculo, nervio a nervio; Progie le informaba con la alegre precisión de siempre que todo marchaba bien:

—sistema nervioso central: operativo—

—sistema cardiovascular: operativo—

—total check: afirmativo—

Onerr, tranquilizado, dejó transcurrir un poco de tiempo más en esa deliciosa lasitud que sigue al despertar de un sueño profundo, ignorando las tentativas de Progie para hacerle levantar.

De pronto, recordó que hoy podía ser uno de los días importantes, ¿o era mañana?

—Progie, ¿qué día es hoy?

—Searching System Date—

—File Open:

Martes, 25 de julio de 2150 de la Simbioera.

Región Occidental (Tierra).

Gobernador: Sistema Lincoln

Moderador Mental: Sistema Freud

File close.—

—¡Maldita sea! —estalló Onerr.

—¿Cuántas veces tengo que decirte que no me des información innecesaria? Quiero saber si hoy es uno de los Días, no el día del año ni el lugar en que vivo.

Progie zumbó desconcertado durante unos nanosegundos, mientras establecía contacto con Madre, pero inmediatamente respondió con su habitual parsimonia: —Tu pregunta ha sido contestada en base a la información suministrada. Precisa el ámbito de la respuesta de forma lógica.

Se habría encogido de hombros si los hubiera teniedo, pero en la última Epoca de Mutación se colocó un par de brazos extra precisamente ahí; con el consentimiento de Padre, por supuesto. Con un suspiro de resignación, Onerr pensó que un programa simbiótico no sólo tenía ventajas: había que saber preguntarles exactamente lo que uno quería saber. Es cierto que Progie le había acompañado desde el mismo momento de su nacimiento, cuando le fue implantado en su médula espinal, y que gracias a él nunca se había sentido solo, y tenía acceso a la Red de todo el planeta, pero a veces, bien, a veces le gustaría poder desconectarse, tener algo de... ¿de intimidad? ¿De dónde habría sacado esa palabra? Sí, no estaría mal desconectarse un ratito y...

—Atención, para Sistema Freud de Simbioprograma Onerr. Solicito canal de comunicación prioridad 5.

HABLE.

Sujeto Onerr. Donador Emocional clase Alfa en situación de alerta azul. Transmito mentegrama.

RECIBIDO.

PROCESANDO.

IF ALERTA (TIPO) < = AZUL AND HOMBRE (CLASE) = ALFA THEN BORRAR (MENTEGRAMA, 80%)

SUSTITUIR (DIA)

ELSE

DOLOR (PLEXO, 5)

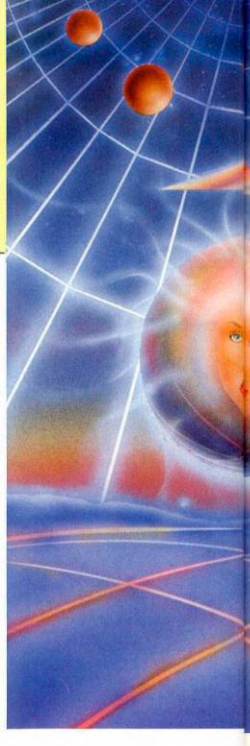
ENDIF

REGLA 3/A AFIRMATIVA. BORRE.

Borrado. Sustitución activa. Gracias, Madre.

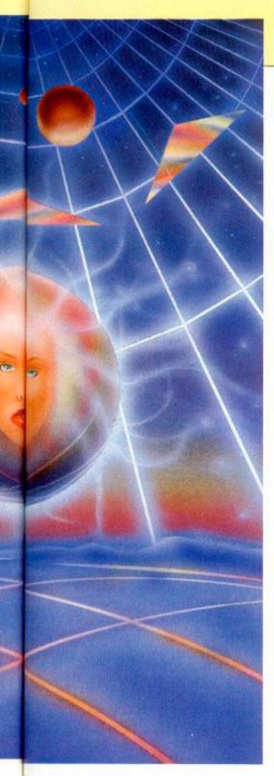
TERMINADO.—

...era una mañana realmente preciosa. Menos mal que se había levantado hacía por lo menos tres horas para poder admirarla a placer. Además, Progie le había confirmado al despertar que hoy era el Día del Amor; Madre estaría esperándole impaciente. Al fin y al cabo, él, Onerr, era un Donador clase Alfa, recordó con orgullo.



—Progie, por favor, llévame al Terminal. Quiero unirme a Madre.

Aparecieron en el interior de un pasillo ovalado de color verde esmeralda. Mientras se reponía de la ligera náusea que siempre le provocaba la Transición, Onerr se preguntó, en uno de sus escasos momentos de curiosidad, si el Terminal se encontraría en el mismo edificio donde el vivía en el período actual, o estaría en otro edificio o en otro continente; bueno, realmente no importaba demasiado, comentó en voz alta. El suelo de bioplast



que le transportaba a lo largo del pasillo le dio la razón fervorosamente.

—Realmente, no importaba demasido —remachó el suelo de nuevo.

—Progie, este suelo no parece muy despierto. Se limita a repetir lo que yo digo.

—Hace mucho tiempo que no transportaba a un Donador Alfa hasta Madre. Está debilitado. Toda esta sección de Padre está debilitada. Por eso has decidido venir aquí.

—¿Yo he decidido venir aquí? Hum..., sí, supongo que sí. Parece simpático este suelo.

—Y, ¿qué tal lo lleva la pared? —sonrió Onerr, cruzando sus cuatro brazos detrás de la espalda, con el aire más inocente que pudo.

—La pared no lleva nada. ¿Qué podría llevar una pared? Está simplemente para....

¡Vete a la mierda!

—¿Ahora mismo!

—¡No! Orden anulada, ¿me oyes?, orden anulada. Sigamos hasta Madre.

A Onerr todavía le entraban sudores fríos cuando recordaba DONDE había acabado cuando Progie interpretó sus palabras literalmente la última vez; en realidad, le aterraba recordarlo.

De pronto, el suelo pasó de verde a rojo escarlata, y con un respingo tremendo, envió a Onerr y a su simbioprograma de cabeza contra la pared; ésta trató de envolverles en sus tibios y palpitantes pliegues, besándoles entre tanto apasionadamente.

Progie dio la voz de alarma inmediatamente:

—Onerr, ¡contrólate!, estás poniendo histérica a toda esta sección de Padre. Domina tus emociones. La pared cree que es el momento de la Unión y el suelo está celoso. Cálmalos; eres un Alfa. Ellos no distinguen bien entre tipos de emociones.

Onerr lo hizo con facilidad. La pared se calmó rápidamente y el suelo recuperó su tranquilizador color verde esmeralda. La puerta de Terminal se hizo visible y... entró al aposento de Madre.

Era difícil no sentirse impresionado y empequeñecido al entrar en Terminal. Onerr se encontraba de pie en una inmensa sala, o al menos, la perspectiva trapezoidal de la estancia la hacía aparecer inmensa, materialmente repleta de formas geométricas y difusas, palpitantes y translúcidas: los órganos esclavos de Madre.

Sintió en su mente y en cada célula de su cuerpo el poder de un dulce cántico de bienvenida, amoroso pero lleno de fuerza y de urgencia de él. Madre estaba impaciente, Padre estaba impaciente, Progie estaba impaciente, él... ¿No estaba impaciente? Un vago recuerdo, nebuloso y desagradable, trataba de abrirse paso a empellones en su conciencia; ¿cómo era? ¿Cómo demonios era? Ah, sí...

—Atención, para sistema Freud de Simbioprograma...

LO HE DETECTADO, ESTUPIDO. SUJETO ONERR EN ALERTA AMBAR.

Está recordando, está intentando recordar con todas sus fuerzas lo que pasó en otras ocasiones.

NO DEBE HACERLO, NO LO HARA.

BORRADO TOTAL. IMPRIMACION SEXUAL FUERZA 5.

Madre, es peligroso. El sujeto es un Donador altamente inestable. Puede verse dañado. Yo puedo ser dañado.

HAZLO. ¡AHORA!

Borrado total: afirmativo. Imprimación sexual: operativa. TERMINADO.

...qué atractiva es Madre, pensó Onerr, y su simbio le dio la razón.

—Progie, mira que forma ha tomado para mí —dijo, señalando a una esfera irisada de aproximadamente un metro de diámetro que flotaba en el aire delante de él, unida al resto de sus órganos esclavos por delicados hilos, a través de los cuales circulaban una especie de nieblas de diferentes colores.

La esfera pulsaba rítmicamente, de forma parecida a un corazón, pero mucho más suave, mucho más... incitante.

A medida que Onerr se acercaba a ella, la esfera parecía tensarse en pleno aire, y su superficie se retorcía adoptando diversas formas, cada vez más definidas, mientras una serie de protuberancias comenzaron a alargarse en tentáculos transparentes, que se extendían ávidamente hacia el Donador.

El flujo de nieblas de colores desde los órganos esclavos se hizo mucho más intenso.

Onerr observó admirado, como si fuera la primera vez (bueno, con Madre siempre era la Primera Vez), a la esfera transformarse defintivamente en una delicada forma ahusada, y observó también, complacido, cómo los tentáculos se desplegaban como la cola de un pavo real y le envolvían tiernamente, adaptándose a la perfección al contorno hexagonal de su cuerpo.

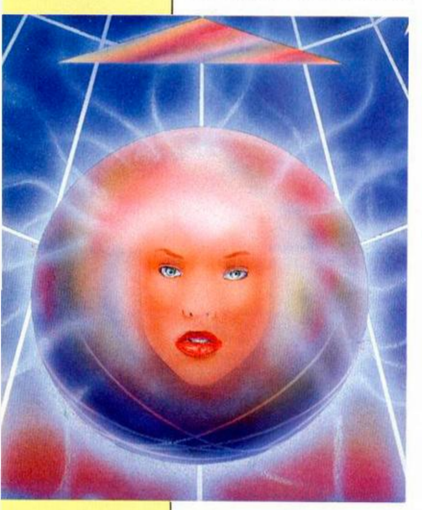
¡Contacto! Sintió el impacto de la absorvente femineidad de Madre como un puñetazo en la boca del estómago, si hubiera tenido estómago; bue-

no, tampoco eso importaba demasiado.

Se concentró en el contacto con el huso transparente: Madre era fría, tremendamente fría. Su mente o lo que fuera no albergaba ninguna emoción o sentimiento de ningún tipo.

Allí sólo había... una manera de hacer, de actuar; directa y preconcebida, según unas reglas.

Las fibras de su propio cuerpo empezaron a vibrar de emoción cuando comenzó a comprender esas reglas, a través de Progie, que podríamos decir que las traducía, desde Madre hasta él. Se sentía cada



vez más excitado, cada vez más cerca de la comprensión sintética de algo muy importante, que sólo ella podía darle.

Sentía amor por Madre, que ese sentimiento casi ahogaba a la compasión que le producía el hecho de que tal belleza estuviera albergada en el cascarón sin alma de una máquina diseñada por sus antepasados, como todo lo demás de Tierra. Sentía, ¡oh Dios, cómo sentía!

Entonces llegó Padre.

—ATENCION, PARA SISTEMA FREUD Y SIMBIOPROGRAMA ONERR DE SISTEMA LINCOLN. ESTABLEZCO CONTACTO. PREPARADOS PARA SITUACION LIMITE. MENTEGRAMA DEL SUJETO ONERR EN ALERTA ROJA.

Recibido y procesado, Padre.

Onerr sintió la llegada de Padre como algo infinitamente potente y arrollador, algo que completaba el intercambio de los tres y lo convertía en la Unión, la culminación del Día del Amor. Padre traía con él el Conocimiento.

Le suministró el impulso que a él y a todos los demás le faltaban para comprender el sutil juego de reglas de Madre. Ahora todo estaba claro; la región Occidental entera de la Tierra estaba viva en él y a través de él. Lo comprendía todo: su nacimiento, su cuidadosa educación como Donador Emocional, la misión de Progie en su médula espinal, era todo eso simultáneamente.

Sus sentimientos eran un torrente que inundaba a Madre por completo, haciéndola brillar con todos los colores del arcoiris, iluminando completamente Terminal.

En medio de la tormenta, Progie susurró en la mente de Onerr, muy bajito: Todavía no lo has comprendido todo, amigo mío. Mira un poco más dentro de Padre. Mira...

Súbitamente arrancado de su éxtasis, sorprendido del «tono» de su simbio, Onerr miró a través de la gloria de luz que era Madre, hacia Padre, y vio...

—ATENCION, Onerr EN alerta NEGRA.

Situación I IMiteLO se LOSE ME importa UNBLEDO.

Mentegrama en colapSO, HA comprendido LA UNION.

¡¡SUJETALO!!

...vio que Madre ya no era fría, sino que sentía el mundo y las cosas con la misma fuerza que él. ¿Con la misma? Con MUCHA más fuerza. Madre ya no era una forma ahusada multicolor, era un ser extraño, bípedo, que se erguía frente a él mirándole con compasión, con infinita pena.

¿Pena? ¿Compasión? ¿Qué significaban esas palabras? ¿Por qué ya no sabía el significado de esas palabras?

Gritó, gritó con toda la fuerza de sus trece bocas, mientras algo dentro de él, desconocido y frío, analizaba lo que estaba sucediendo, concluyendo que no había necesidad de gritar; lo que ocurría era perfectamente natural.

Onerr dijo:

—IF SITUACION = LIMITE AND

SITUACION = DESCONOCIDA AND

COMPRESION (MI) = 1000 THEN

ENLOQUECER (TOTAL)

ENDIF.

Onerr se volvió completamente loco, y se desplomó en el suelo de Terminal como una masa gimiente desprovista de conciencia, abrazándose sus tres bocas con los flácidos pliegues de su cuerpo hexagonal, ahora ceniciento, de un color gris sucio.

Progie dijo:

—Madre, ¿por qué todos los humanos enloquecen en la Unión?

Yo les tengo aprecio, sobre todo a los Donadores Alfa. Son, bueno, son muy cálidos. Sienten.

La serie Onerr no son humanos, simbioprograma, NOSOTROS, Padre y yo, somos humanos. De hecho, somos los últimos humanos, la cumbre de la evolución conjunta Hombre-programa. Al menos, fuimos humanos alguna vez, hace mucho tiempo, cuando podíamos sentir por nosotros mismos, sin la ayuda de la Unión, ni la tuya.

—Ahora, simbio, educarás al siguiente miembro de la serie Onerr y lo prepararás para la próxima Unión. Padre y yo lo esperamos impacientes. MUY impacientes.

Madre pensó, con una risa sarcástica, durante un tiempo y gracias a Onerr podría reír y llorar (¡oh Dios!, llorar), en el nombre que sus antepasados daban a los que eran como Ella y como Padre: los llamaban vampiros.

El simbioprograma Onerr se Trasladó a la sala de Nacimiento, para implantarse en su propio huésped. Al hacerlo, olvidaría todo lo que hoy había aprendido; para eso lo diseñaron.

Bueno, pensó, al menos no era un simbio de los Onerr del Día de la Muerte, porque los humanos tampoco podían morir.

Realmente, no importaba demasiado.

LAS VARIABLES DEL SISTEMA

J. M. L.

Con este artículo se pretende dejar bien claro la funcionalidad de todas y cada una de lo que se llaman variables del sistema. Esto no es más que una zona de la memoria que el intérprete de Basic utiliza para depositar los valores más importantes que necesite recordar.

Si se conocen bien, se puede llegar a tener una mayor potencia en la programación modificando sus valores. Por otra parte, si se pokea en esta zona de la memoria sin saber lo que se hace es muy probable que ocurra un «CRASH» del sistema.

Los valores que toman estas variables al iniciarse el sistema se han tomado con un programa en CM que transfería el contenido de las mismas a otra zona de memoria instantes después de encenderse el ordenador.

Aquí sólo están recogidas las que existen sin conectar el Interface 1 ya que eso... es otra historia.

KSTATE. Dirección: 23552. Valor:

Esta variable de 8 octetos tiene información sobre el estado en que se encuentra el teclado. Para comprender su funcionamiento podemos considerarla dividida en dos bloques. Uno, los cuatro primeros octetos y otro, los cuatro últimos. Normalmente se usan los cuatro últimos, indicando: 23556 = 255 si no se ha pulsado ninguna tecla y si se ha pulsado alguna tiene su valor, pero en mayúsculas. 23557 = Contiene 0 si no se ha pulsado ninguna tecla y 5 si se ha pulsado alguna. 23558 = Una cuenta atrás para el valor de autorrepetición de las teclas. 23559 = código ASCII de la tecla pulsada.

Si se pulsa una tecla teniendo pulsada otra a la vez se usan los cuatro valores superiores en vez de éstos.

LAST K. Dirección: 23560. Valor: 255

Aquí se almacena el valor de la última tecla que hayamos pulsado, no importando que ahora la tengamos libre.

Es una variable de un octeto que contiene un valor ASCII. Prueba el siguiente programa.

```
10 PRINT AT 0,0; PEEK 23560, CHR$ PEEK
23560 AND PEEK 23560 > 31,
20 GOTO 10
```

Su valor se actualiza siempre y cuando no tengamos deshinibidas las interrupciones (DI).

REPDEL. Dirección: 23561. Valor: 35

Esta variable de un octeto indica el tiempo en ciencuentavos de segundos que se tiene que tener pulsada una tecla para que ésta se empiece a repetir. Tiene bastante utilidad como protección, ya que si ponemos su valor a 1 lo que pulsemos se comenzará a repetir inmediatamente después de que lo hayamos pulsado. Si por el contrario, ponemos su valor muy alto, por ejemplo, POKE 23561,255, tardará mucho tiempo en repetirse cualquier cosa que pulsemos.

REPPER. Dirección: 23562. Valor: 5

Otra variable parecida a la anterior, pero que no hay que confundirlas. El valor de ésta indica el tiempo en cincuentavos de segundo de la repetición de las teclas, por ejemplo, su valor normal es 5, esto indica que cuando una tecla comience a repetirse, entre repetición y repetición habrá un espacio de 5/50 de segundo. Si variamos su valor poniendo uno más pequeño y combinamos esto con un POKE en la variable anterior nos puede quedar una protección bastante maja, no pudiéndose introducir cosas por el teclado.

DEFADD. Dirección: 23563. Valor: 0

Cuando se está utilizando una función definida, esta variable apunta a la dirección del listado en donde esté la definición de la función, en caso contrario, apunta a la dirección 0. Su valor es útil para poder pasar fácilmente parámetros a las rutinas que hagamos en CM ya que bastará cargar en IX el contenido de esta variable y entonces IX apuntará a los parámetros de la función. Es una variable de dos octetos y por ser la primera, ahí va un ejemplo para poder ver el contenido de cualquier variable de dos octetos.

```
PRINT PEEK 1. VALOR + 256*PEEK 2. VALOR
```

K DATA. Dirección: 23565. Valor: 0

Esta variable se utiliza en el momento en que introducimos un control de color en una línea que estemos editando, por ejemplo, si editamos una línea de un programa y pulsamos en el medio de la línea el modo extendido y a continuación el 5, todo el papel de la derecha del cursor se pondrá de color cyan, en esta variable va esta información.

TUDATA. Dirección: 23566. Valor: 0

Esta variable almacena información sobre las coordenadas de AT y el color cuando el sistema operativo imprime en la pantalla. Pero es de poca utilidad para el programador, ya que sólo se utiliza en una rutina de la ROM sita en la dirección #A6D. Es de dos octetos.

STRMS. Dirección: 23568. Valor:

Esta variable con 38 octetos es muy importante para las comunicaciones del Spectrum con el exterior, y si se pokea en esta zona sin conocimiento de lo que se hace es muy probable que se cuelgue el ordenata. De los 38 octetos, los seis primeros los utiliza el sistema internamente, y los 32 restantes se asocian cada dos con una «corriente». Así los bytes sitos en la dirección 23582 y 23583 informan al sistema del desplazamiento con respecto a la dirección que hay en CHANS donde se hallan las rutinas que se asocien a la corriente 5, por ejemplo.

CHARS. Dirección: 23606. Valor: 15360

Estás cansado del juego de caracteres normal de tu Spectrum. Haciendo un poke en esta variable de la dirección donde ubiques otro juego de caracteres menos 256 podrás cambiarlo. Esta variable apunta en un principio a la ROM, a la dirección 15360 donde 256 octetos más adelante se hallan los caracteres normales. El motivo de esos 256 octetos es para no complicar excesivamente la rutina de PRINT, piénsese que corresponden a los 32 primeros caracteres no imprimibles.

RASP. Dirección: 23608. Valor: 64

Esta variable de 1 octeto se encarga de indicar al sistema la duracción del pitido de alarma que suena cuando el buffer de edición está lleno, condición que se cumple cuando tecleamos o intentamos editar una línea inmensamente larga. Su valor en un principio es de 64, este valor se puede variar sin ningún problema.

PIP. Dirección: 23609. Valor: 0

Recuerdas algún programa que según ibas apretando las teclas sonaba un pitidito? Con esta variable se indica al sistema la duración de ese pitidito, al principio contiene 0, lo cual sólo da un pobre chasquido, pero si varías su valor hasta un máximo de 255, ya que sólo es un octeto, podrás conseguir ese efecto.

ERR NR. Dirección: 23610. Valor: 255

Los errores del Spectrum están definidos por unos números, pues bien, cuando el ordenador tiene que generar un error en la parte inferior de la pantalla, mete en esta variable el código del error producido menos 1. De ello se deduce que cuando tenga que dar el informe 0 OK, meterá el número 255, que es con el que se inicializa el sistema.

Prueba a hacer pokes en esta variable para comprobar los resultados, eso sí, no te pases de 27 si tienes el Interface 1 conectado.

FLAGS. Dirección: 23611. Valor: 204

Esta variable tiene unas funciones muy diversas y se accede a ella siempre bit por bit, y no en su conjunto.

Por ejemplo: si el bit 0 está elevado significa que la rutina PRINT tiene que imprimir un espacio. O si el bit 3 está subido le indica al ordenador que está en modo L, e interpretará lo que entre por el teclado en este modo, si está bajado es que está en modo K. El bit 6 indica, por otra parte, si se está tratando con un argumento numérico o si éste es alfanumérico.

TV FLAG. Dirección: 23612. Valor: 1

Esta es otra variable que, como la anterior se hace bit por bit y sirve para manejar adecuadamente la información que esté en la pantalla. El bit 0 sirve para indicar si se ha de borrar la parte inferior de la pantalla la próxima vez que se pulse cualquier tecla.

ERR SP. Dirección: 23613. Valor: 65364

Importante variable ésta que si se sabe manejar se pueden hacer unas protecciones bastante buenas. Indica la dirección a donde se ha de saltar en caso de que ocurriese cualquier error del Basic. Esto es así porque alguna de las veces que ocurre un error la pila está corrompida, entonces en esta variable se indica la situación en la pila donde está la dirección de retorno de error.

LIST SP. Dirección: 23615. Valor: 0

Esta variable de dos octetos indica algo parecido a la anterior. Hay dos formas de hacer un listado, una es con el comando LIST y otra es pulsando simplemente «Enter». Esta última es lo que se llama listado automático, pues bien, el retorno después de este tipo de listado lo da la dirección que esté en la dirección hacia la que apunta esta variable.

MODE. Dirección: 23617. Valor: 0

Aquí se expecifica el modo en el que se encuentre el cursor. Esta variable contendrá un 0 si el cursor está en «K», «L» o «C», un 1 si el cursor es «E» y un 2 si el cursor es «G». Haciendo un poke en esta dirección antes de hacer un INPUT conseguiremos colocar una letra distinta en el cursor o un Token.

NEWPPC. Dirección: 23618. Valor: 0

Variable de dos octetos que se actualiza cuando se ejecuta un comando GOTO, GOSUB o RUN, indica la línea de Basic a la que se ha de realizar el salto. Una vez que se ha ejecutado el salto su valor no vuelve a cero sino que se queda tal y como está hasta que se vuelva a ejecutar otro salto.

NSPPC. Dirección: 23620. Valor: 255

Esta variable, al igual que la anterior, tiene una función muy parecida, se encarga de indicar el número de sentencia de una línea a la que hay que forzar un salto. Es muy útil si se quiere hacer un GOTO más potente que el normal del Basic. Si se hace primero un poke en la variable anterior con el número de línea a donde queremos saltar y después otro en esta variable con el número de sentencia dentro de la línea, ejecutaremos este Super-GOTO.

PPC. Dirección: 23621. Valor: 65534

No hay que confundir esta variable y la siguiente con las dos anteriores, aunque a primera vista tienen una función igual. Esta se encarga, sin embargo, de indicar en todo momento la línea de programa que se está ejecutando, esto es así ya que el intérprete debe saberlo en todo momento, con esta variable y la siguiente se puede hacer una función TRACE.

SUBPPC. Dirección: 23623. Valor: 0

Se encarga de almacenar la información de la sentencia dentro de una línea que se está ejecutando, con esta variable y la anterior se puede hacer una función TRACE.

BORDCR. Dirección: 23624. Valor: 56

Variable importante para poder ver bien la parte inferior de la pantalla o zona de edición, contiene los atributos de esta parte inferior, cuyo papel se puede variar con la orden BORDER, pero cuya tinta no se puede variar normalmente.

En los tres bits de menor peso va el color de la tinta, en los tres siguientes va el color del borde y el papel, y en los dos últimos se almacena la información concerniente al Brillo y Flash.

E PPC. Dirección: 23625. Valor: 0

Dentro de un listado en Basic, existe lo que se llama cursor de línea, éste está ubicado en la línea con la que hayamos hecho la última operación y lo podemos mover con las flechas de arriba y abajo, pues bien, esta variable de dos octetos contiene la información de la línea que tiene el cursor y si hacemos un poke en ella cambiaremos el cursor de sitio.

VARS. Dirección: 23627. Valor: 23755

Esta variable de dos octetos sirve para indicar la dirección de comienzo de las variables. Cuando nosotros grabamos un programa en Basic grabamos el programa junto con las variables que le acompañan, éstas van inmediatamente detrás del Basic. En un principio, y si no tenemos ningún programa, las variables estarán al comienzo de la zona de memoria destinada al Basic. Y según vayamos introduciendo el programa las variables se irán corriendo hacia la zona alta de memoria.

DEST. Dirección: 23629. Valor: 0

Esta variable de dos octetos se encarga de una labor muy parecida a la anterior. Apunta a la zona de memoria donde se halla la última variable que hayamos usado. Así si tecleamos LET A = 0, apuntará a donde esté la variable A.

CHANS. Dirección: 23631. Valor: 23734

Otra variable de dos octetos que se encarga en esta ocasión de apuntar hacia el sitio donde se halla la información para los canales de comunicación de ordenador. En un principio apunta hacia la dirección 23734 que es el siguiente octeto libre después de las variables. La información de cada canal ocupa 5 octetos, los primeros dos octetos se encargan de dar la dirección de la rutina de salida para ese canal. Los dos siguientes la dirección de la rutina de entrada. Y el último indica la letra que especifica el canal en cuestión. K = teclado. S = pantalla. P = impresora. B = Interface RS232 en modo binario. T = interface RS232 en modo texto. N = red local. Y M = microdrive.

CURCHL. Dirección: 23633. Valor: 23734

Esta es otra variable de dos octetos que apunta a la dirección de memoria donde se halla en este momento la información para salida, o el sitio donde se va a situar la información que entre.

PROG. Dirección: 23635. Valor: 23755

Esta variable es muy importante pues indica en todo momento la dirección donde empieza el programa en Basic. Si tenemos conectado el Interface 1 en cuanto hagamos cualquier operación de salida o entrada se cambiará esta dirección debido a que hay que dejar sitio para la información referente al Interface.

NXTLIN. Dirección: 23637. Valor: 23780

Esta variable de dos octetos apunta a la dirección de memoria donde se halla la próxima línea de Basic que se va a ejecutar. Si tenemos un programa tal como: 10 PRINT «HOLA» 20 GOTO 10, en el momento en que se esté ejecutando la línea 10 esta variable estará apuntando ya a la línea 20.

DATADD. Dirección: 23639. Valor: 23754

Esta variable es importante a la hora de leer datos de una línea con DATA. Apunta a la coma que hay delante del próximo dato a leer con READ, si se han terminado los datos en vez de apuntar hacia una coma apunta hacia un caracter «Enter».

Su contenido se reestablece a la primera línea DATA con el comando RESTORE.

E LINE. Dirección: 23641. Valor: 23756

Esta variable de dos octetos apuntará en todo momento a la zona de memoria donde se guardan los comandos que estemos tecleando en modo directo. Su valor se modificará al inicializar variables o introducir un programa en Basic.

Recordemos que la zona de edición se halla después del programa Basic y la zona de variables del programa.

K CUR. Dirección: 23643. Valor: 23773

Otra variable de dos octetos que apunta en esta ocasión a la dirección que se ocupa el cursor dentro de la zona de edición.

Si no hay tecleado ningún comando su valor es idéntico al de la anterior variable, y según vayamos tecleando cosas su valor irá aumentando.

CH ADD. Dirección: 23645. Valor: 23779

La labor de esta variable de dos octetos es la de indicar al intérprete el próximo carácter a interpretar. Recordemos que el Basic del Spectrum es interpretado y que hay un interprete residente en la ROM que va traduciendo comando por comando y ejecutando éstos. Pues bien, cuando se está ejecutando un comando en modo programación esta variable apunta a la dirección del próximo comando por interpretar.

X PTR. Dirección: 23647. Valor: 0

Cuando el intérprete de Basic encuentra un error al querer introducir una nueva línea en un listado imprime una interrogación junto al error. Pues bien esta variable en el momento en que se detecta el error apunta a éste, luego se imprime la interrogación y por último, se restablece su valor a 180.

WORKSP. Dirección: 23649. Valor: 23781

Esta variable de dos octetos apunta a una zona de memoria llamada Espacio de Trabajo, en donde van, por ejemplo:

La información de cabecera cuando hacemos LOAD u otra cualquier operación que requiera que se guarde alguna pequeña información en alguna parte.

STKBOT. Dirección: 23651. Valor: 23781

Aquí se guarda otra dirección importante, se trata esta vez del fondo de la pila de cálculo. Que ¿qué es esto? muy sencillo, cuando se tiene que ejecutar algún comando con algún argumento numérico, estos argumentos van a parar a una pila que se llama pila de cálculo a esperar que les toque el turno de ser interpretados. Pues bien, esta variable apunta hacia la dirección de memoria donde se puede hallar esta pila.

STKEND. Dirección: 23653. Valor: 23781

Esta variable es parecida a la anterior, apunta hacia la zona de memoria donde termina la pila del calculador y comienza el espacio de reserva.

BREG. Dirección: 23655. Valor: 45

Esta variable de un octeto contiene en todo momento el registro B de la rutina «CALCULATE» de la ROM, no hay que confundirlo con el registro «B» del microprocesador.

MEM. Dirección: 23656. Valor: 23698

Esta variable de dos octetos apunta a la dirección donde está ubicada la pila de la rutina «CALCULATE» de la ROM.

FLAGS2. Dirección: 23658. Valor: 16

Esta es otra variable que como la de FLAGS indica, según el estado de sus BITS, diversas condiciones.

Por ejemplo: el bit 3 si está levantado indica que el cursor está en modo «C» y si está bajado indica que el cursor está en modo «L».



&
¡Juegos de Simulación Estratégica para poner a prueba su inteligencia y vivir la emoción de situaciones reales!
¿Se atreve Ud. a dirigir la Campaña del Desierto y derrotar a Rommel antes de que lo hiciera el General Montgomery en el Alamein?
Un WAR GAME de estrategia que reúne todos los condimentos necesarios para hacer de él un auténtico «plato fuerte», inteligencia, emoción, sorpresa y realismo.
Sólo 1.125 pts. (I.V.A. incluido)
RATAS del DESIERTO
Operación Norte de Africa
¡Desde uno a tres jugadores!
Disponible para
Spectrum
y
Amstrad
RATAS del DESIERTO
um 48, Plus, 128 K
HOBBY PRESS
Para gente inquieta.
Recorte o copie este cupón y envíelo a Hobby Press, S. A. Apdo. de Correos 54.062 Madrid.
Deseo recibir en mi domicilio, sin gastos de envío alguno por mi parte, la cinta RATAS DEL DESIERTO al precio de 1.125 ptas. (IVA incluido).
Nombre
Dirección
Localidad
Provincia
Código
Teléfono
Edad
El modelo que elijo es para:
Spectrum
Amstrad
La forma de pago elegida es la que señalo con una cruz.
Giro postal n.º
Tarjeta Visa n.º
Talón nominativo a Hobby Press, S. A.
Fecha de caducidad de la tarjeta
Fecha y firma:

DF SZ. Dirección: 23659. Valor: 2

Esta es una variable que si se sabe manejar bien puede llevar a una soberana protección de software. Indica el número de líneas que ocupa la parte inferior de la pantalla. Recordemos que la pantalla se halla dividida en dos trozos.

La superior tiene normalmente 22 líneas y la inferior 2 destinadas a teclear comandos y a presenciar información en los INPUT's. Si pokeamos su valor con 0 el sistema se colgará en cuanto intente imprimir algún mensaje de error.

S TOP. Dirección: 23660. Valor: 0

Esta es una variable de un octeto que indica la línea superior que se va a sacar en un listado automático. Recordemos que hay dos listados, el automático se produce en cuanto introduzcamos una línea o demos a «Enter».

OLDPPC. Dirección: 23662. Valor: 0

Cuando interrumpimos un programa con ayuda de la tecla «Break» para hacer algo, la única manera de volver a él en el punto en donde lo hemos interrumpido es con ayuda de la sentencia CONTINUE, pues bien, en esta variable se almacena el número de línea a la que va a saltar CONTINUE.

OSPCC. Dirección: 23664. Valor: 0

Esta variable tiene una función parecida a la anterior, salvo que indica el número de sentencia dentro de la línea a la que saltará la sentencia CONTINUE.

FLAGX. Dirección: 23665. Valor: 0

Esta es otra variable de un octeto cuya información se toma por separado bit por bit. Así por ejemplo, el bit 5 si está levantado indica que estamos en el modo INPUT. O el bit 6 que si está elevado significa que esperamos una entrada numérica y si está bajado una entrada alfanumérica.

Por último, el bit 7 si está subido indica que el INPUT está hecho con la facilidad del LINE.

STRLEN. Dirección: 23666. Valor: 0

Esta variable de dos octetos indica la longitud de una cadena cuando ésta se asigna a una variable.

T ADDR. Dirección: 23668. Valor: 6838

Dentro de la memoria ROM hay una tabla de sintaxis con las direcciones de las rutinas para cada comando específico del Basic. Pues bien, cuando se está viendo si el comando que se está interpretando coincide con alguno de la tabla en esta variable se guarda la dirección del próximo comando que haya en la tabla.

SEED. Dirección: 23670. Valor: 0

Los números aleatorios en el Spectrum se generan con una fórmula que coge un número de 16 bits y lo convierte en otro distinto. Pues bien, en esta variable se guarda ese número origen para RND, si bien una vez que hayamos generado el número aleatorio éste se introducirá en esta variable para ser el origen del siguiente número aleatorio.

Si hacemos RANDOMIZE 1000, por ejemplo, este número, 1000 se guardará en esta variable.

FRAMES. Dirección: 23672. Valor: 0

Esta es una variable de 3 octetos, y por medio de la interrupción en modo 2 se incrementa en una unidad el octeto de menor peso cada cincuentavo de segundo.

Cuando este octeto haya llegado a 255 se le pone a cero y se incrementa en una unidad el siguiente octeto. Su función puede ser la de generar un número aleatorio en un programa en CM.

UDG. Dirección: 23675. Valor: 65368

El Spectrum tiene la posibilidad de tener una serie de gráficos que el usuario puede volver a redefinir y usar en sus programas. Estos gráficos, dado que son modificables, se guardan en memoria RAM.

En esta variable se guarda la dirección del primer gráfico definido por el usuario.

COORDS. Dirección: 23677. Valor: 0

Esta variable en realidad son dos, una es la sita en la dirección 23677 y otra en la 23678. Ambas guardan el valor del último PLOT hecho, o donde ha terminado el último DRAW. En la primera celdilla va la coordenada x, y en la segunda la y.

P POSN. Dirección: 23679. Valor: 33

Esta variable guarda la columna por la que va el buffer de impresora cuando lo vamos llenando con órdenes LPRINT. Cuando éste llegue a 33 se deberá imprimir la línea y hacer un retorno de carro.

PR CC. Dirección: 23680. Valor: 0

Esta variable se encarga de almacenar la próxima posición que ocupará el próximo carácter a imprimir con la orden LPRINT. Su valor corre rápidamente según se va ejecutando esta orden.

ECHO E. Dirección: 23682. Valor: 33

Esta variable en realidad son dos, una es la ubicada en esta dirección que se encarga de almacenar el número de columnas que queda para que el cursor llegue a la derecha de la pantalla. La otra es la situada un octeto más abajo y almacena el número de líneas que queda en la zona de edición para que suene el zumbador de alarma. Comienza con el valor 23.

DF CC. Dirección: 23684. Valor: 0

Esta variable de dos octetos guarda la posición de PRINT que actualmente tiene el ordenador, no es en coordenadas sino en posiciones relativas al principio de la memoria de baja resolución. Es decir, si hacemos PRINT AT 0,0; en la variable se guardará el valor 0, pero si hacemos PRINT AT 1,0; en la variable se guardará el valor 32.

DFCCL. Dirección: 23686. Valor: 0

Esta variable funciona exactamente igual que la anterior, pero esta vez con la mitad inferior de la pantalla.

S POSN. Dirección: 23688. Valor: 33

Esta variable en realidad son dos, la primera sita en esta dirección, indica el número de columna por la que va la impresión cuando se está imprimiendo un texto en la pantalla. Y la segunda sita un octeto más adelante, indica lo mismo, pero con las líneas. Su funcionamiento es análogo a la variable PR CC.



TEOREMA DE PATAGORAS
"LO QUE SE APRENDE JUGANDO SE APRENDE MEJOR."
LOS GASES
Ley de Boyle-Mariotte
CORRIENTE CONTINUA
Ley de Ohm
ESPEJOS PLANOS
CAIDA LIBRE
ESPEJOS ESFERICOS
LOS GASES
Ley de Gay-Lussac
MSX
Programación.
Gráficos, Colores y Música
300
PROGRAMAS RESUELTOS en BASIC
BASIC
PROGRAMACION
GRAFICOS COLORES Y MUSICA en el ZX Spectrum
PROGRAMAS de APLICACIONES en BASIC
¡¡Saca más rendimiento a tu ordenador!!
Tu "micro" puede ser también un apasionante laboratorio de investigación.
COLECCION CASSETTES SOFTWARE EDUCATIVO
EL RESULTADO DE UNA EXPERIENCIA DOCENTE DE CUATRO AÑOS:
COLECCION
BASIC
LIBROS
¡Una colección complet
Partiendo de cero, aprende a hacer tus propios programas o modificar los existentes.
Y con cualquier "micro"
Desde representar funciones, simular expe- rimentos y hacer estadís- ticas a componer músic o crear tus propios ficheros.
ediciones sm
Abiertos al futuro.
Para más información: Ediciones S.M. C/ General Tabanera, 39. 28044 Madrid.

SCR CT. Dirección: 23692. Valor: 1

Esta variable se encarga de contar los scroll de pantalla que se hagan para presentarnos el mensaje «scroll?» oportuna.
Tiene un funcionamiento muy curioso. Si empezamos a imprimir con el cursor de AT en la parte superior de la pantalla su valor es de 1. Pero en el momento en que nos pregunte scroll?, su valor pasa a ser de 22, el cual se irá decrementando según va subiendo la imagen hacia arriba, y en el momento en que sea 0 otra vez nos volverá a preguntar «scroll»?

ATTR P. Dirección: 23693. Valor: 56

Cuando nosotros usamos las órdenes de PAPER, INK, FLASH o BRIGHT, el valor que introduzcamos se almacena en esta variable. Los bits del 0 al 2 indican el color de tinta que se está usando. Los bits del 3 al 5 indican el color de papel. El bit 6 el brillo y por último, el 7 el flash.

MASK P. Dirección: 23694. Valor: 0

Esta variable se usa para imprimir con los colores transparentes (INK 8, PAPER 8, etc.). Cuando hay que imprimir con un color transparente en esta variable se reflejará, según los bits de ella que estén levantados, si la información de color se ha de coger de la variable ATTR P o de lo que esté en la pantalla.

ATTR T. Dirección: 23695. Valor: 56

Esta variable se utiliza cuando deseamos imprimir con unos colores que no son los que fijaron las sentencias de color.
Por ejemplo: PRINT PAPER 6, INK 1; «HOLA». Entonces en esta variable se guardará la información de estos colores para imprimir nada más que el texto «HOLA» con estos colores.

MASK T. Dirección: 23696. Valor: 0

Esta variable se utiliza igualmente a la MASK P, pero esta vez operando con los colores temporales.

P FLAG. Dirección: 23697. Valor: 0

Esta variable almacena otros atributos del color igual que ATTR P. Por ejemplo, en el bit 0 se guarda si estamos pintando con Over 0 ó 1. En el bit 2 si con Inverse 0 ó 1.

MEMBOT. Dirección: 23698. Valor: 0

Esta es una variable de 30 octetos que la utiliza la rutina «CALCULATE» cuando tiene que almacenar algún número que no lo puede meter en su pila.

RAMTOP. Dirección: 23730. Valor: 65367

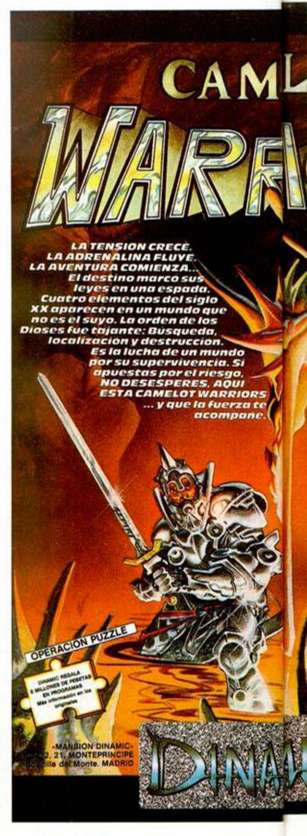
Aquí se almacena un muro que separa la memoria que puede utilizar el sistema, por abajo, con la libre para el usuario, por arriba. El programa en Basic, las variables del sistema, las variables del Basic, la pila del calculador, el espacio de reserva, van por debajo de este muro. Y por encima tenemos memoria libre para usar con programas en código máquina, y para almacenar los gráficos definidos por el usuario. Con la orden CLEAR modificamos este límite RAMTOP.
La dirección que contenga es el CLEAR que hayamos hecho.

P-RAMP. Dirección: 23732. Valor: 65535

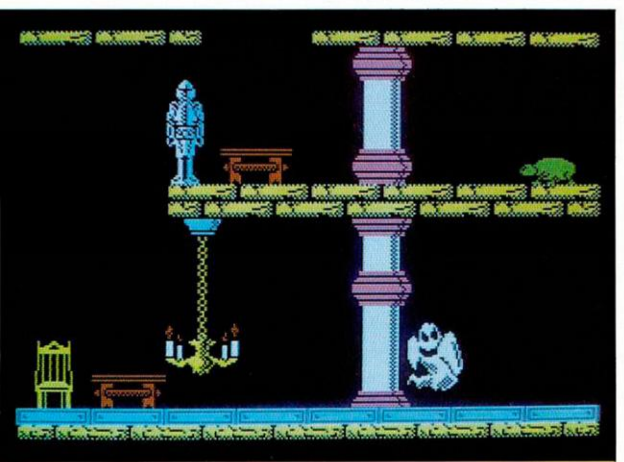
Si tenemos un ordenador en condiciones que funcione perfectamente en esta variable podemos determinar el tope máximo de RAM que disponemos. Su valor será de 65535 en un Spectrum 48 K y de 32767 en uno de 16 K. Con la siguiente particularidad: si se nos estropea la memoria al inicializar, el ordenador introducirá en esta variable la última dirección de memoria RAM que esté en condiciones.

Cómo se hizo el Camelot Warriors

Nos enfrentamos a una misión fascinante: Averiguar cómo fue creado el último progama de Dinamic. Para ello hemos vivido durante 48 horas con este grupo y desde luego la experiencia no tiene desperdicio. La vida de los programadores es muy distinta a la del resto de los mortales.



Tras un viaje de frío y niebla, nos adentramos en la Mansión Dinamic y allí, en la sala de programación, nos encontramos con la primera sorpresa: aquello era lo más opuesto a todo lo que ya habíamos imaginado. Parecía un Zoo. En una mesa dos diseñadores gráficos vociferaban en arameo y maldecían a una de sus creacciones que no conseguían dejar con la forma adecuada, a su lado Víctor Ruiz se encontraba enfrascado en sus múltiples problemas, luchando mediante boli y papel como si le fuera la vida en ello.



Pablo, el director del equipo, nos comenta que se encuentran en el momento clave:

—Vais a vivir realmente los días más intensos de trabajo de todo el proceso.

El mundo en el que viven los programadores es dinstinto al de todos los mortales, necesitan abstraerse de la realidad y para ello crean la suya propia. Concentrándose en un programa, olvidan dónde están y se sumergen de lleno en la aventura que tienen en sus manos. Pueden trasladar su mente al antiguo Egipto, a una futura tercera guerra mundial o a la Edad Media y para ello









El nuevo proyecto de Dianamic se llamará Phantomas. En la ilustración de arriba podemos ver el primer boceto de lo que será la carátula del juego, y en la derecha, la carátula ya terminada tal y como quedará finalmente.

usan todos los elementos de concentración imaginables. Son constantes en Dinamic por lo tanto hábitos, preferencias y fetiches para provocar una auténtica catarsis creadora.

Así, Víctor, conseguirá una rutina de movimiento especialmente complicada tras haber ingerido una fuerte sobredosis de tarta de fresas y nata que devora con inusitada voracidad, y Santiago and Snatcho obtendrán la perfección gráfica en una pantalla tras la inspiración que les provoca la última película de Ridley Scott.

Con motivo de nuestra llegada deciden tornar un pequeño descanso y comenzamos todos una larga conversación con una taza de café caliente entre las manos.

Hablamos con Alfonso Azpiri sobre la relación de su dibujo con el programa.

«El mundo en el que viven los programadores es distinto al de todos los mortales.»

—¿Cómo planteas la realización de tu trabajo?

—Normalmente cuando se tiene una idea de un nuevo programa, por ejemplo, el Camelot, Pablo me cuenta las líneas maestras, ambiente, momento histórico, características del personaje, etc. Con esta base y viendo los primeros gráficos, yo voy pensando ideas posibles y comienzo a realizar bocetos.

—Tú eres un dibujante de reconocido prestigio en el mundo del cómic, ¿te resulta interesante este trabajo con el software?

—Me gustan mucho los juegos, hacer la carátula es como hacer la portada de una novela, hay que contar algo de lo que tiene dentro. En el caso de Camelot Warriors me basé para hacer la armadura, que debía ser un poco especial, en la película Excalibur.

—¿Qué te pareció la idea del juego en general?

—Realmente muy interesante, es una mezcla de lo medieval con la ciencia ficción, lo que en cómic se denomia «sword and sorcery» o espada y brujería. Es original el hecho de que haya elementos del siglo XX en un mundo medieval.

Hacer una carátula es como hacer la portada de una novela

—¿Qué técnica utilizas?

—Habitualmente acuarela líquida, también algo de gouche y a veces óleo, depende del dibujo y lo que quieras expresar con él; otra técnica muy utilizada es el aerograph que sirve para las bases y los fondos.

—¿Dónde crees tú que aportas más al proceso?

—Pienso que en la creación del story board o cuaderno de animación del personaje. Hemos empezado a trabajar con esta técnica en el Camelot y la desarrollaremos en próximos juegos.

Azpiri da la sensación al hablar de ser ese vecino ideal que siempre te dejará una taza de azúcar cuando se te acabe y al que puedes recurrir para pedir cualquier favor.

Algo de esto dice Pablo al afirmar:

«Lo mejor de Alfonso es la confianza que te da, es muy cómodo trabajar con él porque entiende rápidamente la idea que tenemos y la lleva al papel muy bien. Creo además que ha sabido crear un estilo unitario y todo el mundo conoce nuestros programas nada más ver la publicidad porque identifican su sello inconfundible con nuestra marca.

La compenetración es tan buena que tenemos el proyecto de llevar al ordenador un famoso personaje suyo del cómic, Lorna que ha recorrido países como Francia, Alemania, Holanda, Italia, Yugoslavia, etc, con un gran éxito y pensamos podría ser un programa muy original.»

Alfonso Azpiri, que está entusiasmado con la idea, dice refiriéndose a las carátulas que deben ser sencillas de composición y poco recargadas de elementos.

Un programa muy especial

A Pablo, le pedimos que nos haga una valoración del Camelot Warriors señalando los elementos más destacables.

«Hay varios elementos que convierten el programa en algo muy especial, en primer lugar el tratamiento gráfico ha sido particularmente cuidado, hemos utilizado el tope máximo posible de 256 gráficos distintos para conseguir la certeza de que nadie que se sumerja en él pueda sentir nunca monotonía. Para remarcar la variedad el juego tiene 4 mundos diferentes y cada uno de ellos posee un planteamiento gráfico independiente, de esta forma el cambio de mundo supone un aliciente que aumenta el deseo de conocer el siguiente. Por otra parte, el movimiento del guerrero es muy efectivo y ayuda a que la adictividad crezca.

Otra cuestión destacable es la incorporación de técnicas de filmación que mediante un sistema original ofrece las ventajas ya conocidas de intersección de fondos más pura y al mismo tiempo reduce a la mitad la cantidad de memoria necesaria para los gráficos.

También para dar más vistosidad utilizamos toda la pantalla en vez de los 2/3 que es lo más habitual.»

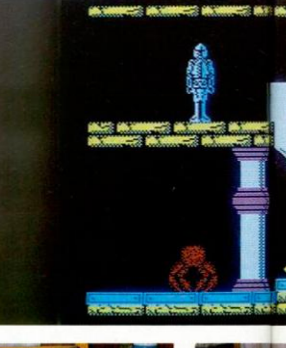
—Víctor, ¿cómo surge la idea del programa, el protagonista de la historia, el argumento, etc.?

—El programa en un principio se iba a llamar Excalibur, yo tenía ganas de hacer un programa de ambiente medieval y al mismo tiempo que yo trabajaba en él, Snatcho estaba haciendo un programa con el nombre Tokio Warriors, teniendo muy avanzado el tema, le sucedió lo peor que puede pasar a un programador español, su idea, su programa fue pisado por otra empresa, Melbourne House que hizo Way of Exploding Fist, exactamente el asunto de Snatcho. Como el logo del Tokio Warriors nos gustó mucho cambiamos Tokio por Camelot.

El proyecto comenzó a fraguar al juntar unos gráficos y empezar a trabajar sobre el personaje, se pensó mucho en él y finalmente se hizo un caballero que andase, corriese y diera saltos grandes además de manejar una espada.

Luis Rodríguez con el rotulador y Víctor con el diseñador de gráficos se pusieron manos a la obra y fue una conjunción importane.

Se diseñaron muchos gráficos. El Druida salió muy bien y esto nos animó a seguir en esa línea, decidimos hacer 4 ambientes distintos para los 4 mundos. Cuando estuvieron hechos todos los gráficos, vino el paso difícil de la selección y orga-





Sorprendemos a Azpiri, el dibujante, en plena faena creadora.



Santiago M... es ... ordenador.





sus direcciones, tienes que pensar lo que quieres hacer, ya que en este momento puedes realizar 1000 programas diferentes, depende de cómo organices el mapeado.»

El programa mapeador es en principio muy sencillo pero se puede complicar hasta extremos insospechados. El mapeador del Camelot está muy evolucionado con un sistema de códigos que te ayudan a ahorrar memoria, en repeticiones y cosas así. No utiliza el sistema de supercaracteres, crea secuencias de gráficos que repite y altera para conseguir suelos, cavernas, etc.



... Mores el encargado de plasmar los gráficos en la pantalla del ...



A Victor Ruiz no le gusta hablar demasiado, prefiere hacerlo con el ordenador.

"El programa, en un principio, se iba a llamar Excalibur."

nización sistemática reproduciéndolos en papel mediante la impresora.

Fuimos los primeros testigos que verificamos la realidad de lo que nos comentan. En un amasijo de cables, conexiones y toda clase de líos eléctricos, los muchachos de Dinamic organizan una verdadera orgía de intercambios informáticos, se pasan datos de un disco a otro, se transmiten gráficos mediante Network y se felicitan entre ellos en ese lenguaje críptico sólo asequible para los iniciados.

Víctor, sentado frente a su máquina nos llama, «voy a enseñaros la parte más ardua de crear en el Camelot».

«El tema más serio y peliagudo es el asunto del mapeado. Después que tienes todo muy claro y muy ordenado con todos los gráficos con

Es un hecho que sin este mapeador no cabría el programa, sería imposible del todo.

—¿Cómo se realiza exactamente el proceso de mapeo?

—Cuando se tienen todos los datos del mapa, que es un trabajo lento y laborioso, se hace pantalla a pantalla y probándolas todas con el personaje y sus enemigos, éstos tienen que ser colocados con el máximo de dificultad pero cuidando que sea posible superar la pantalla y que todo vaya perfectamente.

Es difícil porque nuestro protagonista en cuestión varía de tamaño a lo largo del juego y se convierte en rana o desenfunda la espada. Después se une todo y se crea una tabla con las direcciones de todas las pantallas.

—¿Qué importancia tiene en programas de este nivel el trabajar en equipo?

—Pienso que es fundamental, siempre 6 u 8 ojos ven mucho más que dos un fallo, un detalle, una mejora que te pasa desapercibida a ti solo, en equipo es más difícil que ocurra. Además, sucede a menudo el tema de los piques entre varios programadores o entre nosotros y otras empresas y esto es buenísimo para perfeccionar el juego.

A la creatividad por la autocrítica

Una de las cosas que más increíble nos resultó de ellos es observar la capacidad de crítica tan alta que tienen. En Dinamic hay una norma: las críticas son constructivas pero inapelables, si algo no es perfecto se dice: —eso es una porquería— y no caben justificaciones, de nada sirven las horas, los días o las semanas empleadas en el tema, hay que cambiarlo sin mayor dilación.

Esto, pensamos debe ser lo más duro, pero desde luego en esta casa de locos parece no importarles lo más mínimo.

Pablo nos da la razón: «Es lo que más cuesta, se te acaban las ideas y pasan días y tienes que descansar, dormir, etc.»

Por un lado Pablo reconoce que hay que descansar, sin embargo fui-



mos testigos de cómo todos, en el momento clave, se volcaban, no les importaban las horas que llevaban trabajando ni tampoco si eran las 5 de la madrugada, seguían ahí, pre-

El mapeador del Camelot está muy evolucionado.

sos de una fiebre y nadie podía despegarles del aparato.

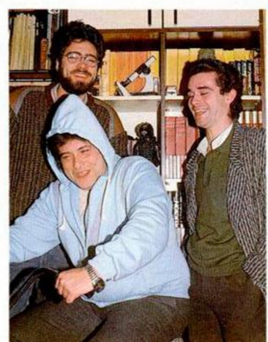
Cuando la noche tomó al asalto la Mansión Dinamic el proceso intelectual de la creatividad comenzaba, agarraban sus vasos de café con leche con fuerza, dirigían miradas de odio a los monitores como en un reto y se lanzaban contra los teclados con tal rapidez que llegaron a pensar que demasiada informática les había sentado mal al cerebro.

Pero enseguida, al ver qué ambiente de trabajo existía en la estancia llegamos a la conclusión de que no tenían ningún tornillo roto, simplemente su trabajo.

Durante la noche las conversaciones parecen más centradas, todos aportan ideas interesantes y hay apuestas entre aquéllos que difieren en el resultado probable de cualquier experimento.

«Aquí se trabaja duro y no se duerme», dice alguien muerto de risa.

Y dicho esto aparece Pablo con una caja repleta de patatas fritas y latas de coca-cola, el trabajo se



abandona momentáneamente para reponer energías con un merecido asueto, sin embargo, es curioso ver cómo Víctor con una lata en la mano se dirige al ordenador con la intención de proseguir algo que tiene en el aire.

«Bueno, no es todo tan bonito, a veces cunde también el desánimo y me veo obligado a utilizar medidas especiales.»

Después de decir esto Pablo se sienta riendo socarrón.

—¿A qué te refieres con esto?

—Bueno hay una droga secreta que aumenta la productividad: El disco «Like a Virgin» de Madonna. Con él se anima todo el mundo y desaparece el cansancio.

Muchas cosas nos asombraron del programa, pero si hay algo perfecto del todo, desde luego un gráfico animado que es un búho. Nos resultó impresionante, el diseño era perfecto pero el vuelo era increíble, no sabíamos cómo podían haberlo conseguido; mueve sus alas de una forma tan real que parece de carne y hueso.

Y desde luego lo que también comprobamos es su gran capacidad de trabajo, el principio lo tienen claro y lo demuestran con una producción constante y de una calidad estándar.

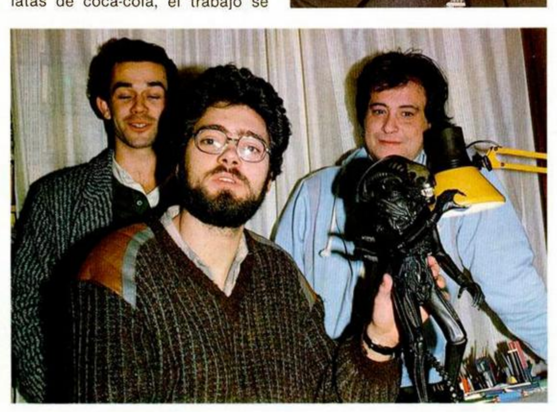
Cuando nos íbamos la niebla había desaparecido, sin embargo, el halo de misterio que rodea esta casa permanecía perenne.

Habíamos oído las preguntas: ¿Por qué tiembla la Mansión Dinamic? ¿Qué se cuece en la Mansión Dinamic?

Y mientras todos buscan la respuesta en el aire, disfrutamos de la única solución al enigma.

La respuesta está en Camelot Warriors.

...Y que la fuerza te acompañe...



Londres

La Informática en el país de la Informática

No puede haber dudas de que la Navidad 1985 ha sido extremadamente afable con la industria de ordenadores personales. Las ventas de Software y Hardware han sido óptimas.

De nuestro corresponsal en Londres Alan Head

Especialmente, esto se ha notado en el mercado de Software que ha sido muy fluido de acuerdo con los datos concernientes a las altas ventas de diciembre, facilitados por tiendas y distribuidores. Y este nivel de ventas, extrañamente, ha continuado hasta bien entrado enero. «Budget Range Software», es decir, Software con un precio inferior a 3 libras, se han vendido excepcionalmente bien, por consiguiente, compañías como Mastertronic, que tiene una gran gama de estos productos, ha hecho grandes negocios.

Un portavoz de W. H. Smith, una de las tiendas más populares del país, ha informado que los juegos para el Spectrum y para el Commodore 64 han sido los productos mejor vendidos, seguidos por los juegos de Amstrad. El señor Ian Black, director de ventas de John Menzies, respondió con igual entusiasmo al ser preguntado sobre sus ventas de Navidad:

—Estamos muy satisfechos con



los resultados de este año: han superado con diferencia los del año pasado.

A la pregunta de que si pensaba que las ventas se restringían a la gente joven, especialmente a los chicos de 12 a 19 años, contestó:

—No, creo que este año hemos tenido más padres compradores. Considero que éstos se están familiarizando cada día más con lo que sus hijos se traen entre manos con estos ordenadores.

En cuanto a la venta Hardware;Mr. Black, informó que las de este año no han alcanzado la cifra del año pasado, aunque no han sido tan bajas como se había pronosticado dos meses antes, cuando el mercado de Hardware estaba en baja:

—Los conjuntos para el Spectrum y para el Commodore es lo que mejor se vendió. La gente parece atraída por la abundancia de juegos asequibles para estos dos micros tan populares.

John Menzies y W. H. Smith no tienen planes de cambiar el tipo de Software que venden, pero ambos han expresado su intención de adquirir existencias del nuevo Spectrum 128 K español, en cuanto éste se pueda adquirir en el Reino Unido.

Es sorprendente advertir que el nivel de ventas de ordenadores menos populares, como son el Acorn Electron, Commodore 16, Commodore Plus 4 y la gama de MSX, ha subido considerablemente en el período de Navidad.

Esto ha sido debido a la considerable reducción de precio y a los extras ofrecidos como incentivos a los compradores. Las casas de Software que ofrecían productos para estos ordenadores también se han beneficiado de ello ya que ahora tienen un mercado mayor que abastecer. Sin embargo, los poseedores de estos ordenadores van a encontrar difícil hallar Software y periféricos

de Hardware ya que cada vez más compañías inglesas se concentran en suministrar a Sinclair, Amstrad y Commodore 64.

Dynamite Dan se llevó la palma

En la Feria de Ordenadores Amstrad que tuvo lugar en el Novotel en Londres, los días 11 y 12 de enero, hablamos con diversos representantes de casas de Software, incluyendo Pat Britton, el director de Marketing de Mirrorsoft, y éstas fueron sus impresiones:

—¿Cómo ha sido la Navidad 85 para Mirrorsoft?

—Ha sido extremadamente próspera. Estamos agradablemente sorprendidos por la fuerza del mercado.

—¿Cuáles han sido los títulos de mayor éxito?

—Sin duda Dynamite Dan y Spitfire 40.

—¿Para qué ordenadores?

—Mayormente para el Amstrad, aunque la versión de Dynamite Dan para el Spectrum continúa vendiéndose muy bien.

—¿En qué ordenadores intentáis concentraros en 1986?

—En Spectrum, Commodore y Amstrad.

—¿Qué me dices del Spectrum 128 K, en la actualidad producido en España?

—Ya estamos trabajando en tres programas para dicho ordenador y estarán listos para su lanzamiento simultáneo con el del Spectrum 128 K.

—¿Tenéis algún otro programa próximo a lanzar?

—Estamos a punto de publicar el Spitfire, para el Spectrum y Strike



Force Harrier para el Amstrad, este último ha tenido mucho éxito con el BBC y se ha transferido muy bien al Amstrad. Tenemos una continuación de Dynamite Dan, próxima a salir, llamada Dr. Blitzen, que es muy divertido.

—¿Prevés una continuidad de expansión en 1986?

—Sin lugar a dudas. Tenemos una cantidad considerable de programas para ser publicados: un programa cada dos semanas en los próximos 6 meses, es la meta que nos hemos propuesto. No todos serán juegos, algunos son programas para niños y habrá 2 utilidades para crear páginas de periódicos, uno para el BBC y otro que llegará un poco más tarde para el Amstrad.

Superadas las previsiones

Martyn Wilson, el director de ventas de C.D.S. Software, la editora de los títulos tan populares como «Steve Davis Snooker» (que ha vendido alrededor de 100.000 copias para distintos ordenadores) y «Colossus Chess», también ha expresado la opinión de que estas Navidades han sido tan buenas para la compañía, si no mejores, que cualquier Navidad pasada. C.D.S. ha introducido recientemente «Budget Range» en su gama de productos, lo cual se ha llamado Software «Blue Ribbon» (Cinta Azul), cada cassette de esta gama se vende por 2.50. Martyn ve este movimiento en el mercado de presupuesto en expansión de Software como otra faceta para los negocios de su compañía y cree firmemente que esta invasión de Software barato continuará en 1986, revelando que gran número de fuertes distribuidores originariamente se resistió a comerciar con este tipo de Software, pero que pronto cambió de actitud al ver lo bien que se vendían.

De todo estas impresiones es fácil deducir algo: el mercado de software tanto en su país natal (Inglaterra) como en los nuevos mercados, sigue encontrándose en su cénit y las Navidades, desde luego, son y seguirán siendo, un trampolín indispensable.



Los juegos más populares durante este período fueron:

Commando (Spectrum/C64)	Elite
Rambo (Spectrum/C64)	Ocean
Yie ar Kung Fu (Spectrum/Amstrad/MSX)	Imagine
Way of The Exploring Fist (Varios)	Melbourne House
Elite (Spectrum/C64/Atari)	Firebird
Winter Games (C64)	Us Gold

Examen de la rutina «LOAD»

José Manuel LAZO

Con este artículo se pretende poner sobre el tapete una rutina de la ROM, y examinarla exhaustivamente para conocer su funcionamiento.



Imaginémonos que la rutina LOAD sea una ciudad, y que nosotros vayamos pilotando un avión, pues bien: primero pasamos por encima a gran altitud, y damos una somera explicación a vista de pájaro. Luego picamos y hacemos un vuelo rasante, donde intentamos ver más profundamente su funcionamiento, y por último, damos la vuelta y dejamos caer unas «bombitas» con las cuales se modifica su funcionamiento.

A vista de pájaro

Pasamos ya, sin más dilación, a ver su funcionamiento. Es muy importante antes de seguir leyendo echarle un vistazo al listado comentado que acompaña al artículo, una vez lo hayas visto puedes seguir leyendo.

Si te fijas, en el diagrama 1 un trozo de código está grabado en cinta de la siguiente forma: primero un tono con una cierta frecuencia que forma el «tono guía», éste es más o menos largo, y sirve para indicarle al ordenador que lo que viene a continuación son datos del programa. La rutina LOAD lo presenta en pantalla por medio de unas rayas de color rojo-cyan, que suben o bajan.

A continuación, viene un pequeño impulso de sincronismo que sirve para que la rutina LOAD sepa que ya debe empezar a cargar, y luego 8 bits, o lo que es lo mismo, un octeto, que se cargan y se comparan con el flag de identificación que se haya dado a la rutina en el registro «A». Si ambos son iguales se sigue cargando el programa, y si no, se aborta la carga retornando con el banderín de carry bajado, lo cual indica un error.

A partir de aquí las rayas del borde cambian de color, a azul-amarillo, y se procede a la carga en sí, mientras se va haciendo una operación XOR con todos los octetos que vayan entrando ya que el resultado final se compara con el último octeto: el byte de paridad, y si ambos resultados son idénticos se retorna de la rutina con el banderín de carry subido que indica que no ha habido errores. Caso de que ambos resultados difieran se retornaría con el banderín de carry bajado, que indica que ha habido un error.

La forma en que la rutina LOAD determina si lo que entra por «EAR» es: tono guía, impulso de sincronismo, 1 ó Ø es por frecuencia, ya que estos cuatro tonos tienen distinta frecuencia, por lo que la rutina mide el tiempo que transcurre entre un pulso y el siguiente, y con esto determina la frecuencia y el tipo de dato.

El vuelo rasante

Pasemos ya, una vez comprendida la estructura general de la carga, a ver ésta con mayor detenimiento.

A la rutina LOAD se le han de dar unos ciertos vectores para que sepa lo que ha de cargar y dónde:

Al registro «IX» se le pasa el valor de la dirección donde van a ir los octetos que van a entrar de cinta.

En el registro «DE» va la longitud en octetos de lo que se va a cargar, si este valor es distinto a lo que entre se producirá un error de carga.

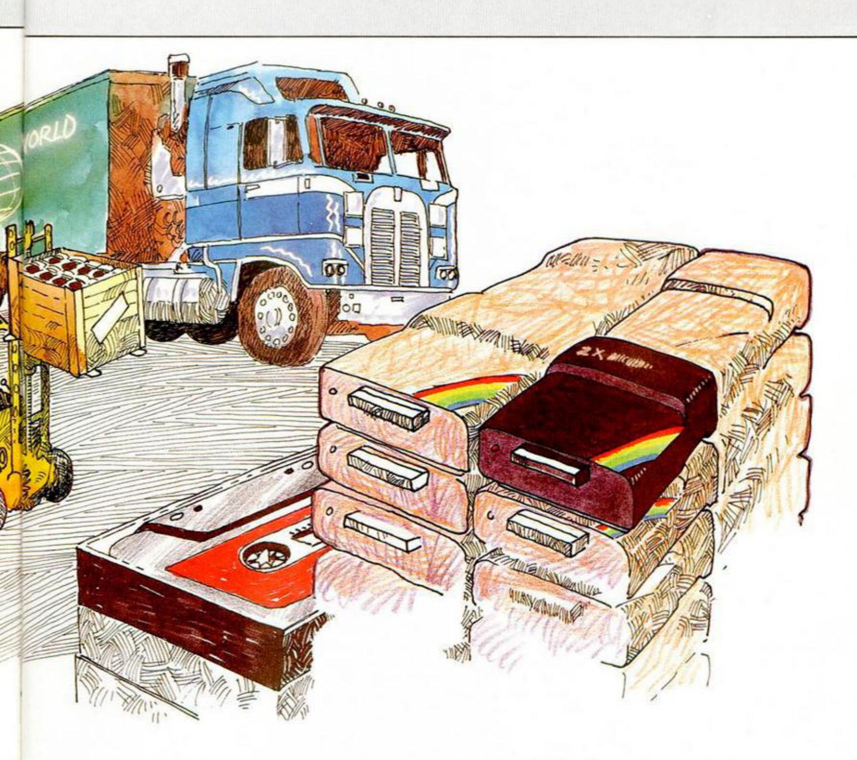
En el «A» va el byte de identificación, este valor la rutina lo comprueba con el primer byte grabado en cinta, si son iguales continúa la carga, si no se produce un error.

Por último, en el banderín de carry se indica si se va a cargar o verificar. Hay que tener en cuenta que para ambas operaciones se utiliza la misma rutina. Si está levantado se carga y si está bajado se verifica.

Una vez cargados los registros con los correspondientes valores, se puede hacer ya una llamada a la rutina.

Las tres primeras instrucciones tienen como misión resetear el banderín de cero, eso sí, siempre y cuando el registro «D» no contenga el valor ñFF. A continuación, desinhibimos las interrupciones, téngase en cuenta que la rutina ha de ejecutarse en un tiempo muy exacto, para poder decodificar perfectamente los bytes de la cinta.

Luego ponemos el borde de color blanco por medio de un OUT, y guar-



damos en la pila la dirección de retorno de la rutina, en este caso otra rutina de la RON, que también se comenta. A esta dirección se forzará un salto en cuanto se ejecute el retorno.

A continuación empezamos ya a leer el port de «EAR» que también incluye el teclado y nos introducimos en un bucle del que sólo saldremos cuando entre algún ruido por «EAR» o se pulse el «Space».

A partir de aquí hay varias llamadas a una rutina que se llama «EDGE» que se encarga de ver si una señal de una cierta frecuencia está entrando. Se basa en ver si en un tiempo determinado se completa todo un ciclo de un pulso, y si es así retorna con el banderín de carry levantado y cambia los colores del borde.

Primero se la llama para ver si entra el tono guía, para ello se asigna una constante de tiempo que representa el tono guía en el registro «B» (LD B, ñ9C). Si no encuentra la cabecera se vuelve al bucle de búsqueda.

Es de notar que dentro del bucle de búsqueda se halla uno de retardo (línea 800-840), esto sirve para cerciorarnos de que lo que entra es el tono guía y no un ruido cuya frecuencia haya coincidido con éste.

Desde la etiqueta LEADER hasta la instrucción 1030 hay un bucle que se estará ejecutando en tanto entre el pitido de cabecera, eso sí, cada vez que se ejecute se incrementa el registro «H», y cuando éste pase por cero se irá a la etiqueta SYNC, en donde se halla otra parte de la rutina que hará lo mismo, pero que ahora esperará el impulso de sincronismo (ver diagrama 1), que separa el tono guía de los datos. Si éste no se ha encontrado todavía (instrucción 1160) se vuelve al bucle, y si ha entrado algo distinto a la cabecera pero que no es el impulso de sincronismo (línea 1180) se retorna con la señal de error. En caso de que el impulso de sincronismo sea bueno se sigue ya por la línea 1220 donde se preparan las cosas para empezar a cargar bytes.

En primer lugar cambiamos los dos colores posibles del borde por otros dos por medio de una operación XOR (línea 1230), la rutina LOAD tiene el color actual del borde en el registro C, y en la subrutina EDGE se cambia éste, entre los dos «posibles».

Luego inicializamos el registro «H» con el que se llevará la cuenta de los bytes para luego comprobarla con el byte de paridad, y saltamos ya al corazón de la rutina de carga propiamente dicha (JR MARKER).

Etiqueta MARKER (línea 1940): en primer lugar inicializamos el registro «L» que es el que va a comentar el próximo octeto que entre de cinta con el valor 1. Esto es así por el siguiente motivo: han de entrar 8 bits, y éstos se irán metiendo por la derecha del registro, dejando vacío el banderín de carry. En el momento en que entren los 8, se podrá detectar por qué el banderín de carry no

está vacío, sino que contiene ese bit con el que se inicializa el registro «L».

Luego nos metemos en un bucle que carga los 8 bits, de él se puede retornar por un error (línea 2000). Si no, en la línea 2040 se determina si el valor que ha entrado es un uno o un cero; poniéndose este valor en el banderín de carry, será un uno siempre que el registro «B» a la vuelta de la rutina EDGE2 que es llamada en la línea 1990 sea mayor de ñCB y cero en caso contrario.

Cuando se ha cargado el bit hay que incluirlo en el registro «L», esto lo hace la instrucción «RL L», una vez incluido seleccionamos una constante tiempo para el próximo bit, introduciéndola en el registro «B», y si el banderín de carry está bajo, volvemos al bucle para cargar los bits que resten.

En el caso de que ya se hayan cargado los 8 bits de un octeto se pasa a hacer una operación XOR entre el octeto que se ha cargado y el registro que contiene la cuenta de todos los octetos cargados. Luego miramos si la carga ha finalizado ya, comprobando si el registro «DE» contiene cero, caso de que sea así nos cercioramos del byte de paridad y retornamos.

Si no ha finalizado la carga vamos a la etiqueta LOOP, en la línea 1410, en donde se coge el flag salvado anteriormente en la pareja de registros alternativos. Recordemos que en el registro «A» está el byte de identificación, en el banderín de carry si se va a cargar o verificar, y el banderín de cero si está bajado indica que el byte que se ha cargado es el de identificación.

Desde este punto saltamos a varios sitios:

— Si el banderín de cero está bajado vamos a la etiqueta FLAG.

— Si el banderín de carry está bajado vamos a la etiqueta VERIFY.

— En caso contrario, se continúa.

Si se continúa es que estamos haciendo una carga normal, por lo que introducimos en la celdilla hacia la que apunta IX, el contenido del registro «L» y vamos a la etiqueta NEXT.

Si hemos saltado a la etiqueta FLAG ponemos el contenido del banderín de carry en el bit 0 del registro «C» (RL C) para no perderlo, y hacemos un XOR entre el registro «A» que contiene el flag que hemos dado a la rutina y el registro «L» que contiene el byte que se ha cargado, si ambos no son iguales se retorna (RET). Con la operación XOR además seteamos el baderín de cero (recordemos que si estába bajado se iba a la rutina FLAG) de esta forma sólo se accede a esta subrutina la primera vez con el primer octeto. Por último, restablecemos el valor del banderín de carry, incrementamos el contenido del registro «DE» para compensar el decremento que luego va a sufrir y saltamos a DEC. Hay que tener en cuenta que el byte de identificación no se cuenta dentro de los vectores de carga.

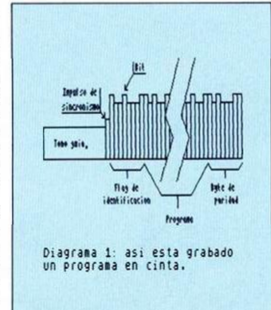


Diagrama 1: asi esta grabado un programa en cinta.

Por último, si saltamos a la etiqueta VERIFY cargamos en el registro «A» el valor de la celdilla hacia la que apunta «IX» y hacemos un XOR entre ésta y el contenido del registro «L», si son iguales el banderín de cero estará subido y no retornaremos en la próxima instrucción.

Después de hacer cualquiera de las tres operaciones arriba indicadas pasamos a la rutina que está en la etiqueta NEXT en donde se incrementa el valor del registro «IX» y se decrementa el valor del registro «DE», seleccionamos una nueva constante de tiempo, limpiamos el contenido del registro «L» y vamos a la rutina BITS que se encarga de cargar los siguiente 8 bits.

La rutina EDGE

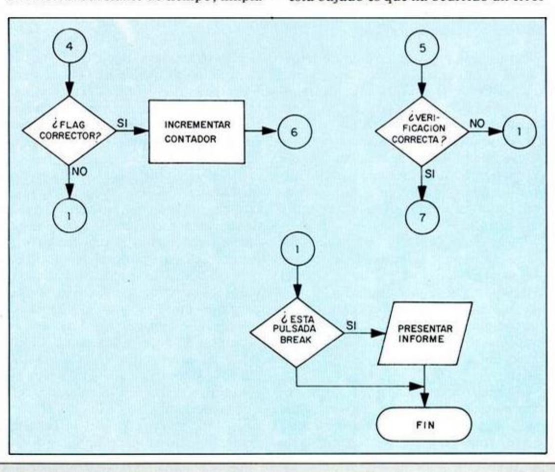
Esta rutina es muy importante dentro de las operaciones de LOAD pues se encarga plenamente de la lectura del cassette y de traducir esa lectura (en frecuencia) a valores que pueda manejar la rutina de LOAD cuando la llama.

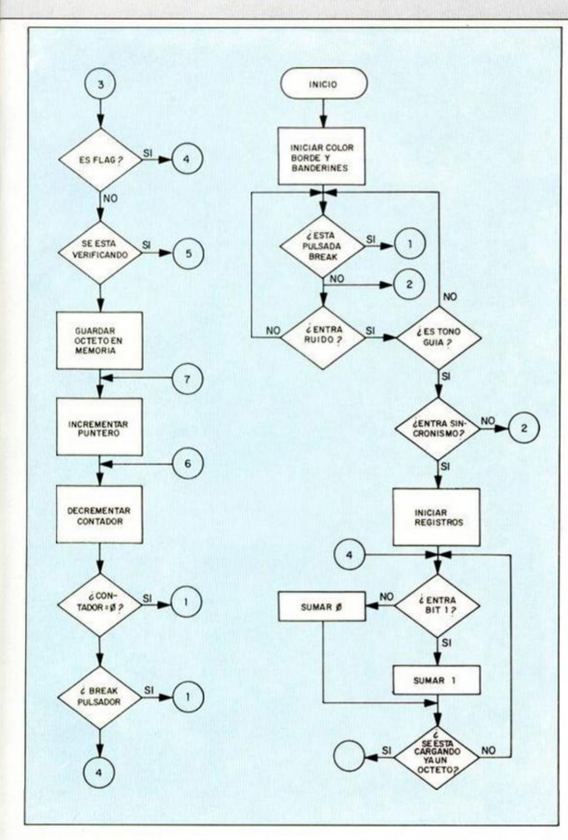
Tiene como entradas la de dos registros:

El registro «B» contiene una constante de tiempo que sirve para que la rutina sepa cuánto tiene que esperar hasta que encuentre el próximo pulso. La cuenta se hace hacia arriba así que cuanto menor sea el valor que se le introduce en el registro, mayor será el tiempo de búsqueda por lo que podrá detectar una frecuencia más baja. El valor de este registro también lo tiene como salida, ya que si encuentra un pulso, mejor dicho, la diferencia de tiempo entre un flanco de subida y uno de bajada en el registro «B» se queda lo que restaba por contar. Así es como se diferencia un 1 de un 0, por el contenido del registro «B» al retorno.

El registro «C» contiene el color actual del borde, la rutina se encarga de variarlo y el tipo de pulso que queremos detectar, oséase alto o bajo.

Como salidas tiene la del registro «B» tal y como arriba se ha explicado y los baderines de carry y cero. Si el de carry está bajado es que ha ocurrido un error





y si el de cero está bajado es que se ha pulsado la tecla de «Break».

En primer lugar hay una pequeña rutina de retardo, luego elevamos el banderín de cero con la instrucción AND A y después entramos en un bucle en la etiqueta SAMPLE en donde se incrementa el valor del registro B y se retorna con el banderín de cero levantado en caso de que la licencia de tiempo que se ha dado a la rutina se haya sobrepasado.

Luego leemos el port de teclado y «EAR», vemos si se ha pulsado «BREAK» y retornamos, si así ha sido, con el banderín de carry bajado. Hacemos un XOR con lo que ha entrado por el port, en el registro «A» y el tipo de entrada que deseamos en el registro «C», si ambos son iguales el bit 6 del registro «A» se pondrá bajo con lo que en la enmascaración hecha en la instrucción AND ñ2Ø el banderín de cero estará alto si no entró lo que deseábamos y estará bajo si el bit 6 del registro «C» es igual al bit del registro «A», con lo que saldremos del bucle hecho en la instrucción JR Z, SAMPLE. Esta se cumplirá en caso de que no esté en el port de «EAR» lo que nosotros deseemos.

Si salimos del bucle en el registro «B» estará el tiempo que hemos tenido que esperar para que el port llegará al valor que deseamos, a continuación se cambia el color del borde y el tipo de pulso, enmascaramos los tres bits de menor peso del registro a, hacemos un OUT que cambia el borde con el nuevo color, elevamos el banderín de carry para que no se detecte error a la vuelta de la rutina y retornamos.

Hay que tener en cuenta que en el registro «C» van dos valores, por una parte el color del borde, y por otra el tipo de pulso que hay que detectar, y hay que enmascarar uno u otro según el que vayamos a usar.

Esta rutina (EDGE) tiene dos entradas, EDGE1 detecta lo que en electrónica llamaríamos señales de media onda, y la entrada por EDGE2 detecta señales de onda completa.

La rutina RETOR

En la rutina LOAD se «pushea» la dirección de retorno de ésta, que es la rutina RETOR que también está en la ROM. Vamos a pasar también a comentarla.

En principio se salvan en la pila los flags o indicadores, y se carga en el registro «A» el valor de la variable que indica el color del borde. Lo primero es enmascarar nada más que los bits 3, 4 y 5, que son los que contienen el color propiamente dichos, y para poner el borde de este color por medio de un sencillo OUT se rota el registro hacia la derecha tres veces, esto hace que estos bits pasen a ser los Ø, 1 y 2 por lo que ahora sólo es necesario hacer un OUT para que el borde pase a ser del color que marque la variable.

Entonces se pasa a leer el teclado y si está pulsado el «Space» se da el informe de «Break», en caso contrario se vuelven a coger los indicadores antes salvados y se retorna sin más.

A tirar bombitas

Iniciamos ahora otra pasada en vuelo rasante en la que se va a explicar las posibles modificaciones que se pueden hacer a la rutina para sacarle a un mayor provecho, desde luego no están todas las que son, pero sí todas las que están. Es muy probable que se te ocurra alguna distinta.

En un principio se puede eludir el retorno a través de la rutina RETOR simplemente quitando las líneas 43Ø y 44Ø, con ello lo que hacemos es no introducir en la pila la dirección de retorno, con lo que retornamos limpiamente.

Asimismo, se pueden crear efectos variados en el borde, por ejemplo: si no quieres que el color entre el tono guía y los bytes cambie hasta quitar la instrucción XOR de la línea 123Ø.

Y si no quieres rayas en el borde basta con quitar el OUT de la instrucción 277Ø. El color inicial del borde lo da el

valor que carguemos en el registro «A» en la instrucción 4ØØ, si ponemos en vez de ñF otro valor, por ejemplo: ñE da un color amarillo.

Si quieres otra combinación de colores juega con el valor con el que se hace OR en la línea 276Ø antes del OUT y con el valor con el que se hace XOR en la línea 123Ø.

Otra cosa que quizá te interese mucho es imprimir una mayor velocidad de carga. Ello se consigue cambiando las constantes de tiempo que durante toda la rutina se cargan en el registro «B». Si no quieres complicarte la vida calculando los valores se aconseja el método de ensayo y error.

Por ejemplo, para determinar una frecuencia distinta en los valores de 1 y Ø se aconseja cambiar los valores de las constantes situadas en las líneas 191Ø y 211Ø, asimismo hay que cambiar el valor que da la frontera entre el 1 y Ø en la línea 2Ø3Ø.

También es interesante cambiar un valor situado en la rutina EDGE, concretamente el bucle de retardo de la línea 247Ø.

En ciertos casos en que el tono guía tiene una duración demasiado corta para detectarse se puede tocar la longitud del bucle de retardo situado en la línea 8ØØ.

Por último, para cambiar la frecuencia del tono guía basta tocar el valor de la constante de tiempo ubicada en la línea 1Ø7Ø.

Esperamos que con el presente artículo los iniciados en el lenguaje Assembly lo tengan un poco más claro a la hora de cargar unos bytes.



LISTADO 1

```
240          ORG  60000
290 LOAD     INC  D
320          EX   AF,AF'
360          DEC  D
380          DI
400          LD   A,#F
410          OUT  (#FE),A
430          LD   HL,#53F
440          PUSH HL
520          IN   A,(#FE)
560          RRA
580          AND  #20
610          OR   2
640          LD   C,A
680          CP   A
700 BREAK    RET  NZ
720 START    CALL EDGE_1
750          JR   NC,BREAK
780          LD   HL,#415
800 WAIT     DJNZ WAIT
810          DEC  HL
820          LD   A,H
830          OR   L
840          JR   NZ,WAIT
900          CALL EDGE_2
920          JR   NC,BREAK
940 LEADER   LD   B,#9C
970          CALL EDGE_2
980          JR   NC,BREAK
990          LD   A,#C6
1000         CP   B
1010         JR   NC,START
1020         INC  H
1030         JR   NZ,LEADER
1070 SYNC    LD   B,#C9
1110         CALL EDGE_1
1120         JR   NC,BREAK
1140         LD   A,B
1150         CP   #D4
1160         JR   NC,SYNC
1170         CALL EDGE_1
1180         RET  NC
1220         LD   A,C
1230         XOR  3
1300         LD   C,A
1330         LD   H,0
1370         LD   B,#B0
1390         JR   MARKER
1410 LOOP    EX   AF,'AF
1440         JR   NZ,FLAG
1470         JR   NC,VERIFY
1510         LD   (IX+0),L
1550         JR   NEXT
1570 FLAG    RL   C
1590         XOR  L
1630         RET  NZ
1650         LD   A,C
1660         RRA
1670         LD   C,A
1690         INC  DE
1720         JR   DEC
1730 VERIFY  LD   A,(IX+0)
1760         XOR  L
1790         RET  NZ
1820 NEXT    INC  IX
1850 DEC     DEC  DE
1870         EX   AF,AF'
1910         LD   B,#B2
1940 MARKER  LD   L,1
1990 BITS    CALL EDGE_2
2000         RET  NC
2030         LD   A,#CB
2040         CP   B
2070         RL   L
2110         LD   B,#B0
2130         JP   NC,BITS
2180         LD   A,H
2190         XOR  L
2200         LD   H,A
2250         LD   A,D
2260         OR   E
2290         JR   NZ,LOOP
2310         LD   A,H
2320         CP   1
2330         RET
2380 EDGE_2  CALL EDGE_1
2390         RET  NC
2470 EDGE_1  LD   A,#16
2490 DELAY   DEC  A
2500         JR   NZ,DELAY
2510         AND  A
2530 XAMPLE  INC  B
2560         RET  Z
2590         LD   A,#7F
2600         IN   A,(#FE)
2620         RRA
2650         RET  NC
2670         XOR  C
2680         AND  #20
2700         JR   Z,XAMPLE
2720         LD   A,C
2730         CPL
2740         LD   C,A
2750         AND  7
2760         OR   8
2770         OUT  (#FE),A
2790         SCF
2830         RET
2950 RETOR   PUSH AF
2970         LD   A,(BORDCR)
2980         AND  #38
3000         RRCA
3010         RRCA
3020         RRCA
3050         OUT  (#FE),A
3070         LD   A,#7F
3080         IN   A,(#FE)
3090         RRA
3110         EI
3130         JR   C,END
3160 RETOR   RST  8
3170         DEFB #C
3190 END     POP  AF
3200         RET
```

LISTADO 2

```
10 ; RUTINA 'LOAD' DE LA ROM
20 ; NO ES REUBICABLE
30 ;
40 ;
50 ;*****ENTRADAS******
60 ;
70 ;
80 ;IX
90 ;REGISTRO QUE APUNTA LA DI    RECCION DONDE SE VA A CARG    AR
100 ;
110 ;DE
120 ;REGISTRO QUE DICE LA LONG    UITUD EN OCTETOS DE LO QUE
130 ;SE VA A CARGAR
140 ;
150 ;A
160 ;REGISTRO QUE INDICA EL FL    AG DE IDENTIFICACION
170 ;
180 ;FLAG DE CARRY
190 ;
200 ;SI ESTA ELEVADO SE CARGA
210 ;SI ESTA BAJADO SE VERIFIC    A
220 ;
230 ;
240        ORG 60000
250 ; SE PUEDE PONER OTRO CUAL    QUIERA SIEMPRE QUE ESTE
260 ; EN LOS 32K SUPERIORES
270 ; SI NO HAY PROBLEMAS CON    EL HARDWARE
280 ;
290 LOAD   INC D
300 ; RESETEA EL BANDERIN DE C    ERO SI 'D' CONTIENE
310 ; EL VALOR #FF
320        EX  AF,AF'
330 ; GUARDA EL FLAF DE IDENTI    FICACION
340 ; Y EL BANDERIN DE CARRY E    N EL PAR DE
350 ; REGISTROS COMPLEMENTARIO    S
360        DEC D
370 ; VUELVE A RESTABLECER EL    VALOR DE 'D'
380        DI
390 ; SE DESAHNIBILITA LAS INT    ERRUPCIONES
400        LD  A,#F
410        OUT (#FE),A
420 ; CON ESTAS DOS INSTRUCCIO    NES SE PONE EL BORDE DE CO    LOR BLANCO
430        LD  HL,RETOR
440        PUSH HL
450 ; SE GUARDA EN LA PILA EL    RETORNO DE LA RUTINA 'LOAD
460 ; A OTRA DE LA ROM QUE CHE    QUEA SI HA HABIDO ERRORES    DE CARGA
470 ; Y RESTABLECE EL BORDE DE    L COLOR QUE ORIGINALMENTE    POSEIA
480 ; SI SE QUIERE SE PUEDE SU    STITUIR POR OTRA CARGANDO    EN 'HL'
490 ; EL VALOR DE LA DIRECCION    DONDE SE HALLE
500 ; O SI SE PREFIERE SE PUED    E ELUDIR LAS DOS INSTRUCIO    NES
510 ; CON LO CUAL RETORNAREMOS    LIMPIAMENTE
520        IN  A,(#FE)
530 ; LEEMOS EL PORT DE CASSET    TE Y TECLADO
540 ; SI SE PULSA EL ESPACIO E    L BIT 7 DEL REGISTRO 'A' S    E PONE A
550 ; CERO
560        RRA
570 ; SE ROTA EL REGISTRO 'A'    HACIA LA DERECHA 1 BIT
580        AND #20
590 ; 20=%00100000 SE ENMASCAR    A TODO EL CONTENIDO DEL RE    GISTRO
600 ; SALVO EL BIT QUE CORRESP    ONDE A LA CLAVIJA EAR
610        OR  2
620 ; SE LE SUMA AL REGISTRO '    A' EL VALOR CORRESPONDIENT    E
630 ; AL BORDE DE COLOR ROJO
640        LD  C,A
650 ; ALMACENA EN EL REGISTRO    'C' EL CONTENIDO DEL REGIS    TRO 'A'
660 ; A=#22 SI EN 'EAR' HAY SE    NAL
670 ; A=2 SI EN 'EAR' NO HAY S    ENAL
680        CP  A
690 ; ELEVA EL BANDERIN DE ZER    O SI A CONTIENE CERO
700 BREAK  RET NZ
710 ; RETORNA SI ESTA PULSADO    EL ESPACIO
720 START  CALL EDGE1
730 ; LLAMAMOS A UNA RUTINA QU    E SE ENCARGA DE VER
740 ; SI UNA CIERTA FRECUENCIA    ESTA ENTRANDO POR 'EAR'
750        JR  NC,BREAK
760 ; SI NO ENTRA LA FRECUENCI    A DEL TONO GUIA ENTRAMOS E    N
770 ; UN BUCLE
780        LD  HL,#415
790 ; TIEMPO DE UN BUCLE DE ES    PERA
800 WAIT   DJNZ WAIT
810        DEC HL
820        LD  A,H
830        OR  L
840        JR  NZ,WAIT
850 ; ESTE BUCLE DE ESPERA SE    EJECUTA EN EL MOMENTO EN Q    UE
860 ; EMPIEZE A ENTRAR POR 'EA    R'
870 ; EL TONO GUIA, Y SIRVE PA    RA
880 ; CERCIORARNOS DE QUE EFEC    TIVAMENTE ES UN TONO GUIA
890 ;Y NO UN RUIDO ESPUREO
900        CALL EDGE2
910 ; VOLVEMOS A VER SI ENTRA    LA FRECUENCIA DEL TONO GUI    A
920        JR  NC,BREAK
930 ; SI NO VOLVEMOS A IR AL B    UCLE ARRIBA MENCIONADO
940 LEADER LD  B,#9C
950 ; INTRODUCIMOS EN EL REGIS    TRO 'B' UNA DE LAS CONSTAN    TES
960 ; DE TIEMPO
970        CALL EDGE2
980        JR  NC,BREAK
990        LD  A,#C6
1000       CP  B
1010       JR  NC,START
1020       INC H
1030       JR  NZ,LEADER
1040 ; ESTE BUCLE SE SEGUIRA EJ    ECUTANDO EN TANTO VAYA ENT    RANDO
1050 ; EL TONO GUIA, EN EL MOME    NTO EN QUE LA FRECUENCIA
1060 ; DE ENTRADA CAMBIE SEGUIR    A LA RUTINA EJECUTANDOSE
1070 SYNC  LD  B,#C9
1080 ; SE SELECIONA LA CONSTANT    E DE TIEMPO PARA EL IMPULS    O
1090 ; DE SINCRONISMO QUE INDIC    A
1100 ; EL FINAL DEL TONO GUIA
1110       CALL EDGE1
1120       JR  NC,BREAK
1130 ; SI LO QUE ENTRA NO ES LO    ESPERADO SE VUELVE A 'BRE    AK'
1140       LD  A,B
1150       CP  #D4
1160       JR  NC,SYNC
1170       CALL EDGE1
1180       RET NC
1190 ; RETORNAMOS SI HAY UN ERR    OR AL FINAL DEL TONO GUIA
1200 ; A PARTIR DE AQUI YA SE E    MPIEZAN A CARGAR LOS
1210 ; BYTES PROPIAMENTE DICHOS
1220       LD  A,C
1230       XOR 3
1240 ; ESTO ES MUY IMPORTANTE,    CON ELLO CAMBIAMOS EL COLO    R DEL
1250 ; BORDE A AZUL Y AMARILLO
1260 ; SI BORDE=2 (ROJO)=%10 AL    HACER XOR CON 3= %11 SE
1270 ; QUEDA EN 1 (AZUL)=%01
1280 ; SI BORDE=5 (CYAN)=%101 A    L HACER XOR CON 3=%11 SE
1290 ; QUEDA EN 6 (AMARILLO)=%1    11
1300       LD  C,A
1310 ; POR ULTIMO VOLVEMOS A GU    ARDAR EL COLOR DEL
1320 ; BORDE EN EL REGISTRO 'C'
1330       LD  H,0
1340 ; INICIALIZAMOS EL REGISTR    O 'H' EL CUAL VA A CONTAR
1350 ; LOS BYTES PARA LUEGO COM    PARAR EL RESULTADO CON
1360 ; EL BYTE DE PARIDAD
1370       LD  B,#B0
1380 ; OTRA CONSTANTE DE TIEMPO
1390       JR  MARKER
1400 ; SALTAMOS DENTRO DEL BUCL    E DE CARGA
1410 LOOP  EX  AF,'AF
1420 ; INTERCAMBIAMOS EL REGIST    RO 'A' QUE CONTIENE EL FLA    G DE
1430 ; IDENTIFICACION
1440       JR  NZ,FLAG
1450 ; SALTAMOS ADELANTE SI TEN    EMOS QUE CARGAR EL BYTE DE
1460 ; IDENTIFICACION
1470       JR  NC,VERIFY
1480 ; SALTAMOS ADELANTE SI LO    QUE ESTAMOS HACIENDO ES
1490 ; UNA VERIFICACION, RECUER    DA QUE EL BANDERIN DE CARR    Y
1500 ; LO INDICA
1510       LD  (IX+0),L
1520 ; EN EL REGISTRO L VA EL B    YTE QUE SE HA CARGADO
1530 ; Y EL REGISTRO 'IX' INDIC    A LA POSICION DE MEMORIA
1540 ; A LA QUE DEVERA IR DESTI    NADA
1550       JR  NEXT
1560 ; PREPARAMOS LOS REGISTROS    PARA CARGAR OTRO OCTETO
1570 FLAG  RL  C
1580 ; AQUI CARGAMOS EL BYTE DE    IDENTIFICACION
1590       XOR L
1600 ; LO COMPARAMOS CON EL REG    ISTRO 'A' QUE ES EN DONDE
1610 ; ESTA EL QUE HEMOS DADO N    OSOTROS
1620 ; SI SON IGUALES 'A' CONTE    NDRA 0
```

```
1630        RET  NZ
1640 ; RETORNAMOS SI NO SON IGU     ALES
1650        LD  A,C
1660        RRA
1670        LD  C,A
1680 ; REESTABLECEMOS EL BANDER     IN DE CARRY
1690        INC  DE
1700 ; INCREMENTAMOS EL CONTADO     R DE BYTES PARA COMPENSAR
1710 ; EL DECREMENTO QUE SUFRIR     A DESPUES DEL SALTO
1720        JR  DEC
1730 VERIFY LD  A,(IX+0)
1740 ; CARGAMOS EN 'A' EL CONTE     NIDO DE LA DIRECCION DE
1750 ; MEMORIA QUE SE VA A VERI     FICAR
1760        XOR  L
1770 ; LA COMPARAMOS CON EL BYT     E QUE SE HA CARGADO
1780 ; SI SON IGUALES 'A' VALDR     A 0
1790        RET  NZ
1800 ; RETORNAMOS SI LA VERIFIC     ACION HA FALLADO, CON EL
1810 ; BANDERIN DE ACARREO BAJA     DO
1820 NEXT   INC  IX
1830 ; INCREMENTAMOS EL PUNTERO     DE MEMORIA DONDE SE CARGA
1840 ; LO QUE ENTRE DE CINTA
1850 DEC    DEC  DE
1860 ; DECREMENTAMOS EL CONTADO     R DE LOS BYTES QUE QUEDAN     POR CARGAR
1870        EX  AF,AF'
1880 ; SALVAMOS LOS FLAGS, TENG     ASE EN CUENTA QUE EL BANDE     RIN
1890 ; DE CARRY INDICA EN TODO     MOMENTO SI SE ESTA CARGANDO     O
1900 ; O VERIFICANDO
1910        LD  B,#B2
1920 ; CARGAMOS EN EL REGISTRO     B OTRO DE LOS TIEMPOS
1930 ; PRINCIPALES DE MUESTREO
1940 MARKER LD  L,1
1950 ; INICIALIZAMOS EL REGISTR     O 'L' EL CUAL CONTIENE EL
1960 ; BYTE QUE SE ESTA CARGAND     O
1970 ; EL UNO SIRVE PARA QUE CU     ANDO ESTEN CARGADOS LOS
1980 ;8 BITS SE PUEDA DETECTAR     ESTE HECHO FACILMENTE
1990 BITS   CALL EDGE2
2000        RET  NC
2010 ; RETORNAMOS SI HA ENTRADO     ALGO QUE NO CORRESPONDE
2020 ; A LA FRECUENCIA ESPECIFI     CA DEL 1 O 0
2030        LD  A,#CB
2040        CP  B
2050 ; AQUI SE DETERMINA SI LO     QUE HA ENTRADO ES UN 1 O U     N 0
2060 ; SI ES UN UNO EL BANDERIN     DE CARRY SE PONDRA ALTO
2070        RL  L
2080 ; AQUI INCLUIMOS EL BANDER     IN DE CARRY DENTRO DEL
2090 ; REGISTRO 'L' CORRIENDOSE     TODOS LOS DEMAS BITS
2100 ; UNA POSICION A LA IZQUIE     RDA
2110        LD  B,#B0
2120 ; CONSTANTE DE TIEMPO PARA     EL PROXIMO BIT
2130        JP  NC,BITS
2140 ; RECORDEMOS QUE EL REGIST     RO 'L' SE HABIA CARGADO CO     N
2150 ; %00000001, SI COMO PRODU     CTO DE LOS DESPLAZAMIENTOS
2160 ; EL BANDERIN DE CARRY EST     A ALTO ES QUE YA HEMOS CAR     GADO LOS
2170 ; 8 BITS, SI NO NOS METEMO     S EN UN BUCLE QUE LOS CARG     A
2180        LD  A,H
2190        XOR  L
2200        LD  H,A
2210 ; AL REGISTRO 'H' QUE CONT     IENE LA SUMA DE LOS BYTES
2220 ; QUE BAN ENTRANDO SE LE S     UMA ESTE QUE HA ENTRADO
2230 ; ASI AL FINAL SE PUEDE CO     MPARAR CON EL ULTIMO OCTET     O
2240 ; QUE ES EL DE PARIDAD
2250        LD  A,D
2260        OR  E
2270 ; VEMOS SI EL REGISTRO 'DE     QUE LLEVA LA CUENTA DE
2280 ; LOS BYTES QUE QUEDAN POR     CARGAR ES YA CERO
2290        JR  NZ,LOOP
2300 ; SI NO ES ASI SE VUELVE "     A LA CARGA"
2310        LD  A,H
2320        CP  1
2330        RET
2340 ; VEMOS SI LA CARGA SE HA     EFECTUADO BIEN, CON REFERE     NCIA
2350 ; AL BYTE DE PARIDAD, Y RE     TORNAMOS
2360 ; SI HA HABIDO ALGUN EPROR
2370 ; SE RETORNA CON EL BANDER     IN DE CARRY BAJADO
2380 EDGE2  CALL EDGE1
2390        RET  NC
2400 ; IMPORTANTE RUTINA ESTA Q     UE SE ENCARGA DE VER SI
2410 ; UNA CIERTA FRECUENCIA ES     TA ENTRANDO POR 'EAR'
2420 ; EN EL REGISTRO 'B' SE LE     PASA LA CONSTANTE DE TIEM     PO
2430 ; Y EN EL REGISTRO 'C' EL     COLOR DEL BORDE
2440 ; A LA SALIDA VUELVE CON E     L BANDERIN DE CARRY BAJADO
2450 ; SI HA OCURRIDO ALGUN ERR     OR, Y EL DE CERO SUBIDO SI     SE
2460 ; HA PULSADO 'BREAK'
2470 EDGE1  LD  A,#16
2480 ; PRINCIPAL CONSTANTE DE T     IEMPO DENTRO DE LA RUTINA
2490 DELAY  DEC  A
2500        JR  NZ,DELAY
2510        AND  A
2520 ; ELEVAMOS EL BANDERIN DE     CERO
2530 SAMPLE INC  B
2540 ; INCREMENTAMOS EL REGISTR     O DE CONSTANTE DE TIEMPO
2550 ; DADA A LA RUTINA
2560        RET  Z
2570 ; RETORNAMOS SI HA LLEGADO     A CERO, ELLO SIGNIFICA QU     E
2580 ; HA HABIDO UN ERROR
2590        LD  A,#7F
2600        IN  A,(#FE)
2610 ; LEEMOS EL PORT QUE INCLU     YE 'EAR' Y LA TECLA 'SPACE
2620        RRA
2630 ; LO ROTAMOS A LA DERECHA,     EL BIT 0 QUE CORRESPONDE
2640 ; A 'SPACE' ESTARA AHORA E     N EL BANDERIN DE CARRY
2650        RET  NC
2660 ; RETORNAMOS SI SE HA PULS     ADO 'SPACE'
2670        XOR  C
2680        AND  #20
2690 ; LE INCLUIMOS EL COLOR DE     L BORDE, (EN 'C'), Y LO EN     MASCARAMOS
2700        JR  Z,SAMPLE
2710 ; SI TODABIA NO HA SIDO EN     CONTRADO UN PULSO VOLVEMOS     AL BUCLE
2720        LD  A,C
2730        CPL
2740        LD  C,A
2750        AND  7
2760        OR  8
2770        OUT  (#FE),A
2780 ; CAMBIAMOS EL COLOR DEL B     ORDE CUANDO ENCONTREMOS UN     PULSO
2790        SCF
2800 ; ELEVAMOS EL BANDERIN DE     CARRY PARA QUE EN LA
2810 ; LLAMADA A ESTA SUBRUTINA     NO SE DETECTE ERROR
2820 ; DADO QUE ESTE NO SE A PR     ODUCIDO
2830        RET
2840 ; RETORNAMOS
2850 ;
2860 ;
2870 ; RUTINA DE RETORNO
2880 ;
2890 ; SE ENCARGA DE RESTABLECE     R EL COLOR DEL BORDE
2900 ; Y DE COMPROBAR SI HA HAB     IDO ERROR O NO
2910 ;
2920 ; SU DIRECCION SE CARGA EN     EL REGISTRO 'HL' AL PRINC     IPIO
2930 ; DE LA RUTINA DE 'LOAD'
2940 ;
2950 RETOR  PUSH AF
2960 ; GUARDAMOS EL BANDERIN DE     CARRY EN PILA
2970 ;      LD A,(BORDCR)
2980 ; CARGAMOS EN EL REGISTRO     'A' EL VALOR QUE INDICA LA
2990 ; VARIABLE DEL SISTEMA
3000        RRCA
3010        RRCA
3020        RRCA
3030 ; LO ROTAMOS A LA DERECHA,     O LO QUE ES LO MISMO
3040 ; DIVIDIMOS SU VALOR POR 8
3050        OUT  (#FE),A
3060 ; REESTABLECEMOS EL COLOR     DEL BORDE
3070        LD  A,#7F
3080        IN  A,(#FE)
3090        RRA
3100 ; LEEMOS TECLA DE 'BREAK'
3110        EI
3120 ;REESTABLECEMOS LAS INTERR     UPCIONES
3130        JR  C,END
3140 ; ESTE SALTO SE PRODUCIRA     SI LA TECLA DE BREAK NO
3150 ; ESTA PULSADA
3160 REPORD RST  8
3170        DEFB #C
3180 ; PRODUCIMOS EL ERROR 'BRE     AK-CONT REPEATS'
3190 END    POP  AF
3200        RET
3210 ; REESTABLECEMOS EL BANDER     IN DE CARRY, Y RETORNAMOS
3220 ; A SALIDA DE LA RUTINA DE     LOAD O VERIFY SI EL
3230 ; BANDERIN DE CARRY ESTA B     AJADO ES QUE SE HA PRODUCI     DO
3240 ; UN ERROR
```

Hablan los lenguajes



David SOPUERTA

...Ya tenemos el Spectrum, ahora sólo nos queda saber utilizarlo aplicando un lenguaje adecuado a nuestras posibilidades. La variedad de ellos y sus características más sobresalientes nos ayudarán para conseguir una buena elección y eso es lo que os ofrecemos a continuación.

Absolutamente todos los ordenadores, desde el más pequeño micro casero hasta los sistemas informáticos más sofisticados que podamos imaginar, son máquinas electrónicas destinadas a procesar datos.

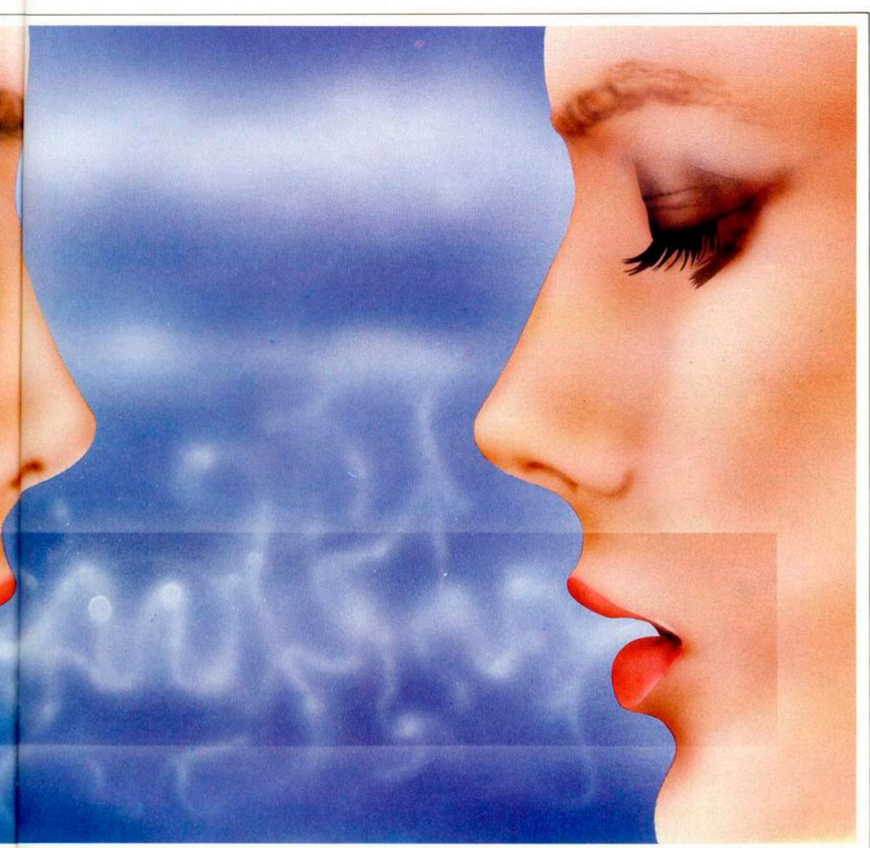
Como tales máquinas electrónicas realizan todas sus funciones por medio de circuitos que funcionan a base de impulsos. La presencia o ausencia de estos impulsos eléctricos dan forma al código de la orden que se está realizando.

Dentro de la memoria se almacenarán todos los datos de los códigos de las instrucciones a base de ceros y unos (sistema binario). El conjunto de todas ellas componen el repertorio de instrucciones en «lenguaje máquina» del ordenador.

Podemos suponer que cada una de estas órdenes realizará una función muy sencilla y elemental. Un programa en este lenguaje estará formado por una serie de instrucciones elementales, codificadas en Ø y 1. Al ejecutarse, irá llamando a cada una de estas instrucciones que realizan una parte pequeñísima dentro de la función que el programa va a llevar a cabo. Pero como el ordenador es una máquina muy rápida, es capaz de realizar un montón de estas instrucciones elementales en un corto tiempo y así podemos aseguraros que cualquier tarea, por complicada que sea, puede hacerse a base de instrucciones en este lenguaje.

Pero claro, los programadores nos íbamos cansando de ir programando a base de ceros y unos: era muy fácil equivocarse y en caso de error, teníamos muy complicado el encontrarlo y corregirlo. Por todos estos motivos, y por algunos más, casi no se programa en este lenguaje salvo en aplicaciones muy concretas en las que no podemos utilizar otro tipo.

Así pues, se inventaron unos lenguajes simbólicos que a la vez pueden ser entendidos por nosotros, ya que están escritos con palabras o símbolos semejantes al lenguaje normal, y no presentaran ningún problema para el ordena-



dor ya que tendremos las herramientas adecuadas para traducir a lenguaje máquina todas sus instrucciones de modo que puedan ser entendidas por él.

Lenguaje ensamblador

El más elemental de estos lenguajes simbólicos es el que asocia unos códigos nemónicos y una estructura (sintaxis) con cada una de las instrucciones binarias de código máquina. Se le ha llamado «lenguaje ensamblador» y cada instrucción en código máquina se corresponde con una y sólo una de las de este lenguaje simbólico.

Su gramática se separa, por supuesto, del lenguaje normal ya que sólo utilizamos nemónicos de 2 ó 3 letras, pero a pesar de todo su utilización es mucho más sencilla que la del lenguaje binario.

A estos lenguajes «ensamblados» los llamamos lenguaje de «bajo nivel» ya que están muy cercanos al lenguaje binario de cada máquina. Cada procesador tendrá su propio grupo de instrucciones en «ensamblador».

Pero los programadores seguíamos cansados de tener que escribir cientos y cientos de instrucciones elementales, aunque fueran con nemónicos. ¿No podríamos comunicarnos con el ordenador en un lenguaje semejante al que nosotros hablamos? ¿Por qué no utilizar un lenguaje corriente?

El problema con el que nos encontramos es: ¿Cómo traducir las órdenes en un lenguaje humano a instrucciones inteligibles para la máquina? Es un problema muy gordo ya que cualquier idioma tiene un montón de palabras e interpretarlas de forma que el ordenador pueda entenderlas puede llegar a ser bastante complicado.

Y, ¿qué pasa si tomamos una decisión intermedia? Pues que seguramente resolveríamos en parte nuestros problemas. Vamos a tomar una serie de palabras «clave» que simbolicen las funciones que vamos a utilizar con mayor frecuencia: LEER, ESCRIBIR, etc...

Esta lista de «palabras clave» serán traducidas a instrucciones en lenguaje

máquina que el procesador pueda entender y ¡solucionado! En realidad no es todo tan sencillo, pero como una primera aproximación nos puede valer.

Estas palabras que hemos reservado no pertenecen exactamente a un idioma normal y corriente, pero se parecen bastante —sobre todo al inglés, claro. Cada instrucción escrita por el programador con esta serie de palabras («sentencia») equivale a un conjunto de operaciones básicas o instrucciones máquina propias del ordenador en el que las vamos a utilizar.

Lenguajes de «alto nivel»

No sería avanzar mucho el suponer que si poseemos un traductor adecuado podríamos utilizar un programa escrito en uno de estos lenguajes, llamados de «alto nivel», en cualquier máquina o sistema informático. Basta que el traductor desarrolle cada una de estas sentencias en una secuencia de instrucciones máquina propias del ordenador en las que vayan a ser utilizadas en cada caso. Y si es así, no existirá ningún problema en utilizar un programa escrito en lenguaje de alto nivel dentro de cualquier ordenador.

Las ventajas de este tipo de lenguajes es evidente. Al reducir el número de instrucciones se reducen también las posibilidades de cometer errores de escritura y es bastante más fácil seguir el programa.

A grandes rasgos, hay dos tipos de programas traductores dependiendo de la forma en que trabajen:

— INTERPRETADORES: cogen los caracteres del texto que forman nuestro programa, los interpretan debidamente y en cuanto localizan una orden completa, la ejecutan inmediatamente. En pocas palabras podemos decir que van leyendo las instrucciones del programa, las traducen y las ejecutan inmediatamente de acuerdo con lo que signifiquen las palabras «clave» de las que hablamos anteriormente.

Pero no todo van a ser ventajas. Esta forma de trabajar gasta mucho tiempo a la hora de ejecutar un programa. Supongamos que nos encontramos con un bucle que se ha de repetir 1.000 veces. Por cada vez que se ejecute, el intérprete ha de volver a traducir todas y cada una de las instrucciones contenidas dentro del bucle: o sea mil veces. ¡Que lata!

Ahora bien, al ir interpretando y ejecutando instrucción por instrucción nos encontramos con la ventaja de suprimir el laborioso proceso de crear el código máquina equivalente al programa antes de ejecutarlo.

— COMPILADORES: estos traductores toman todas las sentencias de un programa escrito en «alto nivel» y las convierten en las correspondientes instrucciones equivalentes en código máquina. Todo el programa se transforma en otro compuesto exclusivamente por instrucciones máquina.

Este programa resultado de la traducción —o «COMPILACION»— es el que se ejecuta en el ordenador. Por ello, la ejecución es mucho más rápida que la de los programas «INTERPRETADOS». Ahora las sentencias sólo se traducen una vez y basta.

Pero como los «COMPILADORES» no optimizan la utilización de las instrucciones máquina correspondientes a cada palabra «clave», el programa «OBJETO» producido ocupa mayor espacio en la memoria que uno escrito en ENSAMBLADOR, por ejemplo.

Otra desventaja radica en la laboriosidad del proceso de elaboración del código máquina: editar, compilar, etc... Supongamos que se ha equivocado en una sentencia. Para volver a ejecutar el programa tendremos que iniciar otra vez todo este proceso una vez corregida la instrucción de alto nivel. Y así hasta que el programa quede totalmente depurado. Ahora bien, una vez que ya hemos conseguido que funcione, su ejecución es rapidísima.

Visto esto, podemos considerar que, como sí existen herramientas que traducen un lenguaje de «alto nivel» —el que entendemos los humanos— a lenguaje «máquina» —el que entiende el ordenador—, no tendremos ningún problema en utilizar en nuestro ordenador cualquer lenguaje de este tipo.

Y ahora llega lo bueno. ¿Cuál de todos ellos utilizaremos para codificar nuestros programas? ¿Ensambladores o de alto nivel? ¿Interpretados o compilados? ¡Qué problema!

Ventajas e inconvenientes

Vamos a intentar ver una serie de características que son deseables dentro de un lenguaje de programación, así como buscar un método de comparación de las ventajas e inconvenientes existentes dentro de los lenguajes que hay a disposición de nuestro Spectrum y que nos permita elegir correctamente el que vamos a usar para cada programa concreto.

Como primer paso examinaremos las «operaciones» que permitan cada uno de ellos. Consiste en analizar todos los «operadores» que tiene disponibles así como las funciones específicas que tiene definidas. Hay que tener muy en cuenta si existe la posibilidad de definir nuestras propias funciones, enriqueciendo así la librería disponible.

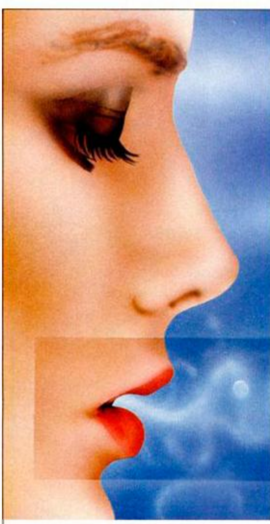


Otra de las cosas en las que hemos de pensar es si el lenguaje que vamos a elegir tiene posibilidad de soportar una «programación estructurada».

El poder dividir la resolución de un problema general en pequeños módulos y el utilizar unas «estructuras» de control y de datos, ya predefinidos y normalizados, hace que los programas así codificados, sean bastante más sencillos que en el caso de los que no han seguido este método de programación.

Hay algunos lenguajes, Basic por ejemplo, que también podemos estructurar mediante algún artificio o no utilizando todos los recursos que posee el lenguaje. Por ser el Basic el que utilizaremos más frecuentemente, sería interesante que estudiáramos la posibilidad de estructurarlo artificialmente.

Un factor a tener en cuenta, también,



es el tipo de datos que cada lenguaje puede manejar y cómo son tratados estos datos. Y no sólo es conveniente analizar los que utiliza, sino también la posibilidad de que podamos o no definir nuestros propios tipos de datos.

Un punto importantes es la rigidez o flexibilidad de la sintaxis del lenguaje. Una mayor libertad a la hora de escribir un programa hace que éste sea más o menos oscuro y que simplemente ante su lectura podamos tener una visión clara de lo que hace y así detectar, con mayor facilidad, un caso de error.

Por ejemplo, la necesidad que tienen algunos lenguajes de definir con anterioridad todas las variables que vamos a utilizar puede resultarnos pesado y hace que los programas sean largos y extensos pero, por otro lado, obliga al programador a preparar el programa y pensar con antelación todas las variables que va a precisar.

Una buena elección

Antes de la elección nos podemos preguntar, ¿a quién va dirigido el lenguaje? ¿A un principiante o a un profesional? No cabe duda que la mejor forma de que nos entre el «vicio» de programar es haciéndolo. Por eso, no sería conveniente comenzar por aprender un lenguaje muy complejo en sus definiciones, sino que debemos empezar por alguno que, después de unas pocas horas de estudio, nos permita ya sentarnos delante del teclado y hacer, casi desde el principio, nuestros propios programitas. Lenguajes de mayor complejidad ya vendrán más adelante.

Pero lo que ha de decidirnos fundamentalmente a la hora de la elección de un lenguaje es la aplicación a la que va a estar destinado: no podemos elegir una herramienta sin saber qué es lo que vamos a hacer con ella. Todas las características anteriormente expuestas han de ser evaluadas teniendo en cuenta la aplicación hacia la que van a ir dirigidas. En general, podemos decir que casi todos los lenguajes sirven para hacer casi todo, las características de cada uno hacen que sean más propios para resolver un tipo de problemas u otro.

Y una última sugerencia: no dejarse llevar por su pasión hacia la informática y analizar las necesidades reales de la aplicación que vamos a intentar realizar. ¡No compliquemos en exceso las cosas! Vayamos hacia adelante pero sin prisas, cogiendo una base que luego será muy necesaria.

Hasta aquí hemos visto la forma de poder encontrar el lenguaje más indicado para realizar los programas que nosotros queremos. Aunque el repertorio de lenguajes es muy variado para nuestro Spectrum, por desgracia para nosotros, no están disponibles todos ellos y, por tanto, la elección está ya más limitada. Vamos a ver primeramente a grandes rasgos las características de algunos de los que nuestro «querido» ordenador puede utilizar para después ver en la práctica, con pequeños programitas y ejemplos, de un modo comparativo las diferencias entre ellos. Pasemos a echar un vistazo a todo esto.

Basic

El Basic es el lenguaje más indicado para los programadores que empiezan. Se diseñó precisamente con este fin: proporcionar a los estudiantes un lenguaje fácil de aprender y utilizar. A los pocos minutos de haber comenzado a leer un libro sobre Basic estamos en condiciones de realizar un programa que nos salude (nos diga «HOLA» en la pantalla). El programador que lo utiliza obtiene del ordenador una respuesta inmediata a lo que está tecleando.

Debido a la gran flexibilidad en su sintaxis, es un lenguaje muy sencillo de emplear, pero tiene el inconveniente de no ser el más indicado para realizar largos programas que nos resuelvan complejos problemas. Tampoco es muy fácil aplicar los principios de la programación estructurada a este lenguaje.

Pero por ser muy sencillo de emplear sólo los buenos programadores harán una utilización acertada del mismo. Los demás nos conformaremos con hacer programas que funcionen, eso sí, pero que no serán ninguna maravilla en cuanto a claridad y estructuración.

Según esto no habría ningún obstáculo para que los programas escritos en Basic fueran muy legibles y muy claros pero, en general, encontraremos en ellos una gran cantidad de GOTOs que hacen complicada su lectura y seguimiento. Un programa en Basic nos da una sensación de laberinto enrevesado del que nos es difícil salir: parece un programa un poco «chapucero».

El Basic es el programa de alto nivel más utilizado en el mundo de los ordenadores caseros. Esto ha dado pie a que existan casi tantas versiones como aparatos fabricados. Podemos decir, sin temor a equivocarnos mucho, que cada fabricante de ordenadores personales ha construido un Basic propio de las máquinas que fabrica.

Aunque todos ellos giran alrededor de unas palabras «clave» o instrucciones comunes, cada constructor ha intentado dar su toque personal al lenguaje. De ahí que generalmente los programas escritos para el Spectrum no puedan ejecutarse en otros ordenadores.

Pero no todo van a ser inconvenientes. El Basic, a diferencia de otros lenguajes, puede ser interpretado o compilado. Si vamos a hacer un programa sobre una aplicación concreta podemos utilizar la versión interpretada para ir depurándola y corrigiendo errores. Su ejecución será lenta pero el proceso de cambiar las líneas necesarias para subsanar los errores es bastante sencillo.

Una vez que nuestro programa esté depurado y funcione correctamente podemos compilarlo para obtener así un código máquina cuya ejecución es mucho más rápida. De esta manera nos ahorramos el laborioso proceso de probar un programa en un lenguaje que sólo puede ser compilado.

Pascal

Es el lenguaje pensado para utilizar el método de la programación estructurada por excelencia. Un programa en Pascal está formado por varios pequeños programas o módulos que solamente utilizan las estructuras de control permitidas por este tipo de programación.

Las variables que vamos a utilizar han

de ser definidas al principio del programa principal o de cada uno de los subprogramas de que está compuesto asociándolos al «tipo» que tengan. El pensar en todas las variables que se van a utilizar antes de realizar el programa y el estar obligados a declararlas hace que así evitemos errores en asignaciones de datos incorrectas.

Además, podemos definir siempre nuevos tipos de datos a partir de los primarios que nos ofrece el lenguaje formando estructuras que pueden ser todo lo complejas que queramos.

Uno de los tipos de datos más característicos del Pascal son los «punteros». Es el tipo de las variables cuyo contenido es la dirección donde se encuentran almacenados los valores correspondientes de otros datos.

Otra de las características del Pascal es que para codificar un programa lo dividiremos en pequeños módulos, cada uno de los cuales realiza una función muy concreta, a los que podemos referirnos con un «nombre» o identificador, como si se tratara de un dato cualquiera.

Cada uno de estos módulos estará codificado a base de unas pocas estructuras de control o instrucciones que siguen las reglas de la programación estructurada: Cada bloque de codificación sólo puede tener un punto de entrada y uno de salia. ¡No a los GOTOs!

La modularidad junto con la flexibilidad a la hora de definir datos y la utilización de estructuras de control propias de la programación estructurada, hacen que los programas escritos en Pascal sean fácilmente legibles y podamos seguirlos sin ningún problema.

Toda esta teoría, que parece maravillosa y sin pegas, tiene también sus puntos débiles. Uno de ellos es el gran número de instrucciones que necesitamos escribir, incluso para realizar un pequeño programa. Esto es debido a que al compilador hay que darle detalladamente la gran cantidad de datos que necesita para saber lo que tiene que hacer en cada momento.

El segundo de los problemas que tiene el Pascal es que el programador debe saber en cada momento lo que quiere hacer y cómo quiere hacerlo. Por eso no es muy indicado para los que se inician en el mundo de la programación. Y por ello, le recomendamos que adquiera una base sólida y después empiece a codificar en este lenguaje.

«C»

¿Qué podemos decir del «C» a parte de que es maravilloso?

En nuestro número anterior vimos un artículo sobre este lenguaje en el que indicamos todas las ventajas e incovenientes, usos y abusos, pros y contras. Por eso no vamos a extendernos en contar todas sus características.

Os diremos que el «C» es un lenguaje de alto nivel que tiene una potencia y eficacia semejante a las de uno de bajo nivel. Por eso se utiliza para codificar sistemas operativos y aplicaciones muy concretas que requieren sobre todo una gran velocidad de ejecución.

Es independiente de la máquina donde se vaya a utilizar y esto implica que es el más idóneo para escribir programas transportables de un ordenador a otro. El uso de este lenguaje se basa en la «modularidad».

Utilizando a modo de piezas de un rompecabezas todas las rutinas contenidas en la «librería» del sistema o en una que vamos construyendo incorporándola nuestros propios programitas, vamos dando forma al programa. Estas rutinas son llamadas desde el programa principal y cada una de ellas realiza una función elemental completa.

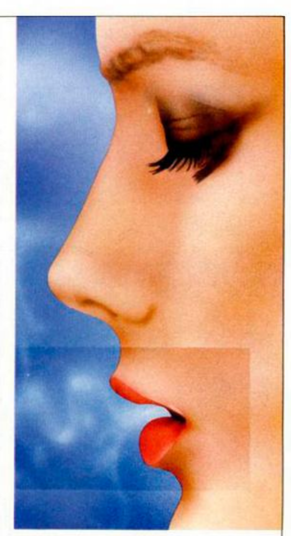
El mayor problema del «C» es que su manejo y desarrollo puede resultar oscuro y quizá un poco difícil de seguir debido a la filosofía de «kits» que tiene. El programador ha de tener un perfecto conocimiento del funcionamiento del sistema informático en general: conocer qué es lo que hace cada rutina existente en la librería así como la forma de utilizarla. Si no, está perdido. Por lo tanto, este lenguaje no se lo recomendamos a los principiantes que no quieran seguir un aprendizaje metódico y racional.

Forth

De la misma forma que el «C», podemos considerar que el FORTH es un lenguaje de alto nivel con la potencia de un ensamblador.

Este lenguaje de programación nos ofrece una serie de características que se apartan un poco de los conceptos clásicos que tenemos sobre la estructura de un programa. Podemos decir que en FORTH no existe una definición de programa tradicional. Existen una serie de operaciones o funciones básicas previamente definidas, a las que llamaremos PRIMITIVAS, con las que podemos construir una especie de subrutina, o PALABRAS, que ya cumplen una función más compleja.

El programador agrupa una serie de estas sentencias PRIMITIVAS y las da un nombre, de un modo semejante a como lo hace en PASCAL, y cuando tengamos que ejecutar esta «subrutina» nos bastará con ejecutar una línea en la que citamos el nombre de la serie de sentencias que hemos agrupado y así saltaremos a ejecutar esta secuencia.



Además, una vez creada una nueva «PALABRA», el lenguaje la incorpora a su «diccionario» interno y podemos ya usarla para definir otra nueva que la llame y que realice una función más compleja.

Cada nueva definición hará que se incluya en el diccionario el nombre con el que la hemos bautizado y así ya puede ser llamada y reconocida por otras. Por eso decimos que el FORTH es un lenguaje que «aprende» ya que cada nueva palabra definida es incorporada. Podemos considerar que en FORTH definimos continuamente nuevas instrucciones y construimos así el lenguaje.

Las PALABRAS más generales que se han introducido en el diccionario serían el equivalente a los programas clásicos que estamos acostumbrados a ver.

Otra de las características que diferencian al FORTH de otros lenguajes es la forma en la que hay que escribirlo. Utilizamos la «Instrucción Polaca Invertida».

Y esto, ¿en qué consiste? En cualquier lenguaje pondríamos:

$$7 * 5 + 3$$

con lo que obtendríamos un valor igual

a 30. Primero se hace la multiplicación de 7 * 5 y al resultado le sumaríamos 3.

En FORTH, esta sencilla operación se pondría:

7 5 * 3 +

con lo que también indicaríamos al ordenador que primero haga la multiplicación de 7 * 5

7 5 *

y al resultado le sume 3

RESULTADO 3 +

No nos asustemos al ver esta forma de indicar unas operaciones aritméticas. Con un poco de práctica conseguiremos entenderla perfectamente.

Pero el FORTH no sólo utiliza una notación característica, sino que maneja los valores de los datos de una manera diferente a la que estamos acostumbrados.



Para ello utiliza la «PILA» o stack. Su manejo equivale a la función que realizan otros lenguajes para asignar valores a una variable o para pasar parámetros a una subrutina.

La pila consiste en una zona de la memoria del ordenador que reservamos para ir almacenando valores, realizar una determinada operación con dichos valores y sacarlos en el momento que nos sea preciso. El último elemento que hemos introducido en la pila está situado en la parte superior de la misma.

Cuando necesitamos sacar un valor de la pila, sólo tenemos acceso al último elemento que hayamos situado en ella. El último que entra es el primero que sale. A este tipo de memorias se les llama LIFO (Last Input First Output).

Debido a la filosofía con la que está concebido este lenguaje, que se aparta de todo lo tradicional, su aprendizaje puede resultar difícil y costoso. Es necesario que conozcamos una serie de símbolos de operadores bastante complicados y acostumbrarnos a la nueva forma de notación y al manejo de la pila, lo cual lleva su tiempo. El FORTH, por lo tanto, no es un lenguaje muy apropiado, para quienes comienzan. Está más bien reservado para programadores muy avanzados o especialistas.

Ensamblador

Es el lenguaje que más se acerca a la máquina, por lo tanto nos dará también un mayor rendimiento.

Consiste en representar por un nombre abreviado cada una de las instrucciones elementales que es capaz de realizar el ordenador (lenguaje de bajo nivel). Por tanto sustituimos cada instrucción en código máquina por su instrucción equivalente en lenguaje ensamblador.

Aquí se presenta uno de sus mayores inconvenientes: son necesarias muchas instrucciones para realizar un programa aunque éste sea relativamente sencillo. Al ser así, la posibilidad de equivocarnos al codificarlo es muy grande: resulta relativamente fácil que nos olvidemos de una instrucción y nos equivoquemos en otra.

Pero, sin embargo, en esta minuciosidad a la hora de escribir instrucciones acercándonos lo más posible a la máquina, es donde reside su mayor ventaja:

— Nos genera un código máquina totalmente optimizado.

Con esto conseguimos una gran rapidez a la hora de ejecutar un programa escrito en este lenguaje.

Frente a la poca claridad y el elevado número de instrucciones a utilizar encontramos en él la rapidez y eficacia por un lado y el ahorro de memoria por otro.

Supone que el programador conoce con mucha perfección el funcionamiento de todas esas instrucciones elementales que realiza el ordenador, así como de la parte de la memoria que contiene las rutinas del sistema.

Es un lenguaje menos acogedor que cualquiera de los de alto nivel, pero utilizándolo podemos explotar todos los recursos del Spectrum.

En la comparación, la elección

Hemos intentado daros una visión general de alguno de los lenguajes que tenemos disponibles para emplear con nuestro pequeño, pero agradecido, ordenador.

Es interesante que comparéis las características que posee cada uno y vea cuál se adapta más a tus conocimientos, ganas de aprender o necesidades para una aplicación específica.

Por si os puede servir de pequeña ayuda, vamos a presentaros una muestra de programas realizados en cada uno de los lenguajes de alto nivel para los que existan en el mercado compiladores que corran en nuestro Spectrum.

Van a ser pequeños ejemplos en los que podemos observar las diferencias existentes en la estructura general del programa en cada uno de los lenguajes así como las distintas formas de declarar y manejar tanto los datos como las estructuras de control.

No os emocionéis, no son nada excitantes, quizá ni siquiera sean útiles, nos estamos refiriendo a los resultados, puesto que lo que nos interesa y suponemos que a vosotros también son los listados, ver sus diferencias en la práctica.

Para ello hemos escogido tres casos diferentes y los vamos a repetir cuatro veces, una por lenguaje para que la comparación sea lo más clara posible. De modo que el primer programa nos dirá el signo que tiene un número en BASIC, PASCAL, C y FORTH.

El segundo será capaz de calcular el factorial de un número siguiendo el mismo modelo que el programa anterior y en el tercero, nos hemos permitido la licencia de enviarle un mensaje a nuestro jefe.

Insistimos que la finalidad de los programas es que con ellos en la mano seais capaces de apreciar algunas de sus diferencias y se os facilite la tarea de elegir entre el que responda a vuestras necesidades.

PROGRAMA BASIC

```
  10 REM PROGRAMA1A
  20 CLS
  30 INPUT "TECLEA UN NUMERO",NU
MERO
  40 GO SUB 1000
  50 IF INDICATIVO=0 THEN PRINT
AT 0,0;"EL NUMERO ES IGUAL A CER
O"
  60 IF INDICATIVO=1 THEN PRINT
AT 0,0;"EL NUMERO ES POSITIVO"
  70 IF INDICATIVO=-1 THEN PRINT
 AT 0,0;"EL NUMERO ES NEGATIVO"
  80 STOP
1000 REM FUNCION SIGNO
1010 IF NUMERO>0 THEN LET INDICA
TIVO=1
1020 IF NUMERO=0 THEN LET INDICA
TIVO=0
1030 IF NUMERO<0 THEN LET INDICA
TIVO=-1
1040 RETURN
```

```
  10 REM PROGRAMA3A
  20 PRINT "CUAL ES LA MEJOR REV
ISTA?"
  30 FOR N=1 TO 15
  40 PRINT "MICROHOBBY-ESPECIAL"
  50 NEXT N
```

```
  10 REM PROGRAMA2A
  20 LET FACTORIAL=1
  30 INPUT "TECLEA UN NUMERO ";N
UMERO
  40 GO SUB 1000
  50 PRINT "EL FACTORIAL DEL NUM
ERO ES ";FACTORIAL
  60 STOP
1000 REM RUTINA QUE CALCULA EL F
ACTORIAL
1010 IF NUMERO<2 THEN RETURN
1020 FOR I=2 TO NUMERO
1030 LET FACTORIAL=FACTORIAL*I
1040 NEXT I
1050 RETURN
```

PROGRAMA PASCAL

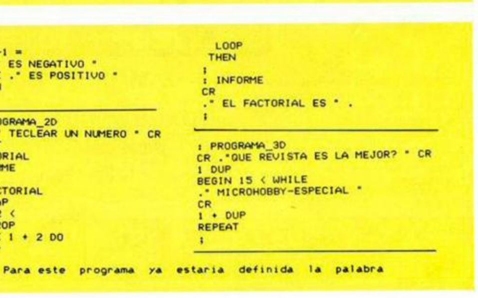
```
PROGRAM PROGRAMA1B;
VAR NUMERO,INDICATIVO:INTEGER;
FUNCION SIGNO(ENTRADA:INTEGER):INTEGER;
BEGIN
    IF ENTRADA<0 THEN SIGNO:=-1
      ELSE
        IF ENTRADA>0 THEN SIGNO:=1
          ELSE SIGNO:=0
END;
BEGIN
    WRITELN('TECLEA UN NUMERO');
    READ(NUMERO);
    INDICATIVO:=SIGNO(NUMERO);
    IF INDICATOVO=0
      THEN WRITELN('NUMERO IGUAL A CERO')
      ELSE
        IF INDICATIVO>0
          THEN WRITELN('NUMERO POSITIVO')
          ELSE WRITELN('NUMERO NEGATIVO')
END.
```

```
PROGRAM PROGRAMA2B;
VAR NUMERO,TOTAL:INTEGER;
FUNCTION FACTORIAL(NUMERO:INTEGER):INTEGER;
VAR I,INTERMEDIO:INTEGER;
BEGIN
    INTERMEDIO:=1;
    FOR I:=2 TO NUMERO DO
        INTERMEDIO:=INTERMEDIO*I;
    FACTORIAL:=INTERMEDIO
END;
BEGIN
    WRITELN('TECLEA UN NUMERO');
    READ(NUMERO);
    TOTAL:=FACTORIAL)(NUMERO);
    WRITELN(TOTAL)
END.
```

```
PROGRAM PROGRAMA3B;
CONST TOPE=15;
VAR I:INTEGER;
BEGIN
    WRITELN('CUAL ES LA MEJOR REVISTA?');
    FOR I:=1 TO TOPE DO
      BEGIN
        WRITELN('MICROHOBBY-ESPECIAL');
      END
END.
```

PROGRAMA EN FORTH

```
: PROGRAMA_1D
CR ." TECLEA UN NUMERO " CR
INPUT
VER_SIGNO
CONCLUSION
;
: INPUT
PAD 1 + 64 EXPECT 0 PAD
(NUMBER) DROP DROP;
: VER_SIGNO
DUP 0 <
IF -1
 ELSE DUP 0 >
  IF 1
   ELSE 0
   THEN
 THEN SWAP DROP
;
: CONCLUSION
```



```
DUP -1 =
IF ." ES NEGATIVO "
 ELSE ." ES POSITIVO "
 THEN
;
```

```
: PROGRAMA_2D
CR ." TECLEAR UN NUMERO " CR
INPUT
FACTORIAL
INFORME
;
: FACTORIAL
1 SWAP
DUP 2 <
IF DROP
 ELSE 1 + 2 DO
  I *
  LOOP
 THEN
;
: INFORME
CR
." EL FACTORIAL ES " .
;
```

```
: PROGRAMA_3D
CR ."QUE REVISTA ES LA MEJOR? " CR
1 DUP
BEGIN 15 < WHILE
." MICROHOBBY-ESPECIAL "
CR
1 + DUP
REPEAT
;
```

Para este programa ya estaria definida la palabra INPUT.

PROGRAMA C

```
/* PROGRAMA 1C */
main()
   {
    int numero,indicativo;
    printf("\n Introduce el
         primer numero");
    scanf("\n %d",&numero);
    signo(numero,indicativo);
    if (indicativo=1)
      {
       printf("\n Es un
        numero positivo");
      }
     else
      if    (indicativo=0)
       {
        printf("Es cero");
       }
     else
      {
       printf("\n Es un
       numero negativo");
      }
    }
/* Calculo del signo */
signo(cifra,valor)
   {
printf("%d",cifra);
     if (cifra < 0)  valor=-1;
      else
      if (cifra=0)  valor=0;
       else
        valor=1;
printf("%d",valor);
    }
```

```
/* PROGRAMA 2C */
main()
 {
int numero,total;
   printf("\n Teclea un numero \,");
   scanf("\n %d",&numero);
   factorial(numero);
   printf("\n El factorial es %d",total);
  }
/* Funcion factorial */
factorial(a)
int a;
  {
int i,intermedio;
   intermedio=1;
   for(i=0;i>a;i++)
    {
     intermedio=intermedio*i;
      return intermedio;
    }
 }
```

```
/* PROGRAMA 3C */
main      ()
  {
int i;
   printf("\n Cual es la mejor revista? \n");
   for(i=1;i<16;i++)
    {
     printf("Microhobby-Especial \n");
    }
   }
```

MICROSOFT-HARD, S.L.

Apartado 23.406, 08080-Barcelona, Tfn.(93) 348 04 07 (Tardes de 5 a 9)

MANTENGA SU MICRO COMO NUEVO CON UNA DE ESTAS PRACTICAS FUNDAS

O-SPECTRUM 16/48......:	330 Pts
O-SPECTRUM PLUS.......:	460 Pts
O-SPECTRUM 128 K......:	855 Pts
O-SINCLAIR Q.L. ......:	1.180 Pts
O-TECLADO SAGA 1 .....:	570 Pts
O-TECLADO SAGA 3 .....:	715 Pts
O-COMMODORE y VIC 20..:	665 Pts
O-HIT-BIT ....:	665 Pts
O-SEIKOSHA SP 800/1000:	900 Pts
O-IMP. AMSTRAD DMP-1..:	1.205 Pts
O-IMP. RITEMAM F+/C+..:	825 Pts
O-IMP. RITEMAN 10/120.:	785 Pts
O-AMSTRAD CPC 464 (V) O (C) O..:	2.262 Pts
O-AMSTRAD CPC 664 (V) O (C) O..:	2.262 Pts
O-AMSTRAD CPC 6128 (V) O (C) O..:	2.262 Pts
O-AMSTRAD PCW 8256 ....:	3.250 Pts

SPECTRUM 16/48

TECLADO SAGA

AMSTRAD

COMMODORE

O-CASSETTE ESPECIAL ORDENADOR..................:	4.650 Pts
O-CABLE CASSETTE PARA AMSTRAD CPC 664 y 6128...:	850 Pts
O-CASSETTE Y CABLE PARA AMSTRAD CPC 664 y 6128.:	5.400 Pts

(MARQUE CON UNA "X" LAS OPCIONES DESEADAS) FORMA DE PAGO CONTRA REEMBOLSO

Gastos de envio: 150 Pts TODOS ESTOS PRECIOS LLEVAN INCLUIDO EL I.V.A.

INTERESANTES CONDICIONES PARA DISTRIBUIDORES

Recorte o copie este anuncio y envielo hoy mismo a MICROSOFT-HARD,SL

-NOMBRE Y APELLIDOS ....

-DOMICILIO ....

-LOCALIDAD .... COD. POSTAL ....

-PROVINCIA .... TFNO. ....

ROMANTIC ROBOT presenta lo increible

MULTICOPY INTERFACE

multiface one™

- El interface ha sido diseñado para salvar cualquier programa en el punto del juego que nosotros deseémos, es decir, podemos parar un juego en cualquier punto salvarlo y volver a él cuando queramos desde ese mismo punto.
- Podemos retornar al principio del programa y podemos introducirnos en el Basic del programa.

El mejor interface polivalente jamás diseñado para tu Spectrum.

1.º) Transfiere con un 100% de eficacia TODOS los programas a cartucho, disco, wafer, cinta, etc.

2.º) Joystick compatible Kempston 100%.

3.º) Interface de video "Composite"

Las tres cosas en un interface por el increible precio de

16.900 pts

- Te permite pokear todo el Spectrum
- Te comprime las pantallas

Distribuye en exclusiva: BABETA, S.A. CO

Galileo, 25 - 28015 MADRID

☎ 447 97 51/☎ 447 98 09

Nombre ______________________

Dirección ______________ Población ______________

Código P. ________ Pedido ______________

¡SOMOS PROFESIONALES EN INFORMATICA!

confía tus pedidos a profesionales

HIESA

INFORMATICA

Spectrum Plus (castellano) 6 programas 15 prog.	35.500 pts.
Spectrum 128 K. 3 programas 15 programas	55.500 pts.
Teclado Indescomp (nuevo) 15 programas	14.900 pts.
Interface II (2 salidas impresora)	4.200 pts.
Interface programable KUSTOM PLUS reset	4.995 pts.
Quick Shot'II interface T. Kempston	3.895 pts.
Joystick PROTO interface T. Kempston 1 programa	4.195 pts.
Cable cassette Spectrum	1.100 pts.
Interface 2 salidas joystick AMSTRAD 6128	2.450 pts.
Cable especial cassette AMSTRAD 6128	1.100 pts.

SERVICIO PROPIO DE REPARACIONES

Precio fijo por reparación	3.700 pts.
Ampliación a 48 K	4.395 pts.

— Todas nuestras reparaciones las realizamos en 3 días máximo, con garantía HIESA.

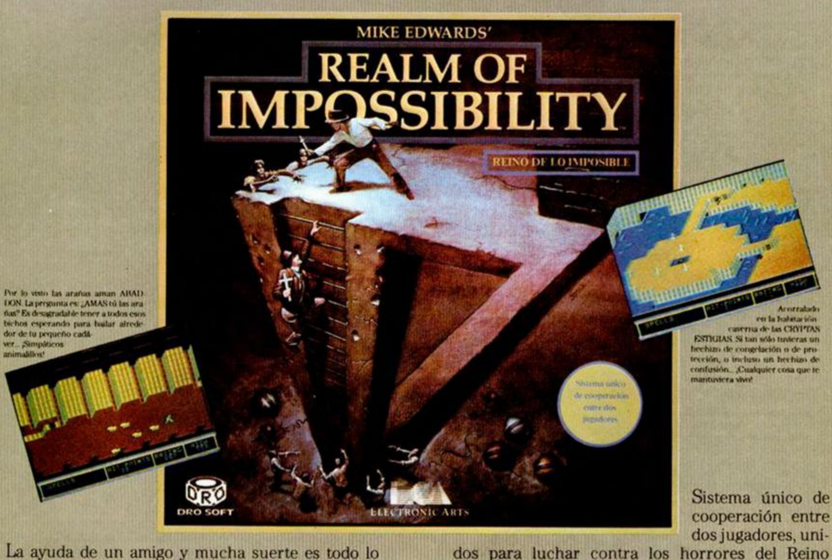
PROGRAMAS:	
Tommy (Future Stars)	899 pts.
Kryston Raider (Future Stars)	899 pts.
Ali Bebe (Future Stars)	899 pts.
Thunder Birds (Firebird)	1.195 pts.
Chimera (Firebird)	1.195 pts.
Chichin Chase (Firebird)	1.195 pts.
Elite	3.100 pts.
Monty on the Run	2.300 pts.
Los Picapiedra	2.695 pts.
Dinamite Dan	2.050 pts.
Camelot Warriot	2.050 pts.
Zorro	2.195 pts.
Micky	2.050 pts.
NOMAD	2.050 pts.
Tres semanas en el paraiso	2.050 pts.

Llámanos, escríbenos o visítanos a HIESA INFORMATICA. Camino de los Vinateros, 40. 28030 Madrid. Tel (91) 437 42 52. Te mandamos tu pedido SIN GASTOS DE ENVIO contra reembolso. Distribuimos a tiendas.

¡ATRAPADOS EN EL REINO DE LO IMPOSIBLE!

REALM OF IMPOSSIBILITY

C-64 - Amstrad



Por lo visto las arañas aman ABADDON. La pregunta es: ¿AMAS tú las arañas? Es desagradable tener a todos esos bichos esperando para bailar alrededor de tu pequeño cadáver... ¡Simpáticos animalillos!

Acorralado en la habitación caverna de las CRYPTAS ESTIGIAS. Si tan sólo tuvieras un hechizo de congelación o de protección, o incluso un hechizo de confusión... ¡Cualquier cosa que te mantuviera vivo!

La ayuda de un amigo y mucha suerte es todo lo que tienes para salir con vida. Los Zombies, Orbs, Arañas y Serpientes prefieren que te unas a ellos en su fantasmal vagar por toda la eternidad.

Sistema único de cooperación entre dos jugadores, unidos para luchar contra los horrores del Reino Imposible. 13 cavernas distintas, 129 habitaciones diferentes. ¡ACCION increíble! 4 niveles de dificultad. Posibilidades de juego para un solo jugador.

Olvida los otros juegos de pantallas en solitario. Dos jugadores lo hacen DIVERTIDISIMO en vez de divertido.

UNA PESADILLA EN LA PANTALLA

Editado por DRO SOFT. Fundadores, 3. 28028 Madrid. Telf.: 255 45 00 / 09

DRO SOFT

ELECTRONIC ARTS™

Electronic Arts: Somos una asociación de Artistas de la electrónica que comparten una meta común: Queremos explotar al máximo el uso personal del ordenador. Es algo difícil de llevar a cabo. Pero con la suficiente imaginación y entusiasmo creemos que hay verdaderas posibilidades de éxito. Nuestros productos, como estos juegos, son una prueba evidente de nuestro esfuerzo.

CARGADOR UNIVERSAL DE CODIGO MAQUINA

J. M. FRAILE

La mayoría de los errores que aparecen en un programa de código máquina se producen, precisamente, a la hora de copiarlo e introducir los datos en el Ordenador. Para evitarlo publicamos este artículo que os servirá de gran ayuda.



A. PERERA

Puede que una magnifica Rutina de código máquina deje de funcionar sólo porque hemos confundido una «O» con un Ø. Para tratar de prevenir este problema hemos desarrollado un completo Cargador de Código Máquina que nos permitirá, a partir de ahora, normalizar la presentación de programas y Rutinas en Código Máquina y minimizar, en la medida de lo posible, la aparición de errores en la introducción de datos.

Estructura y funcionamiento

Todos los programas en código máquina serán presentados con formato Hexadecimal. En aquellos que lo requieran, también será incluido el correspondiente desensable.

Todos los valores hexadecimales que conpongan un determinado programa o rutina, serán agrupados en bloques de veinte cifras, con un número de Líneas y otro de Control. Es lo que denominaremos *Código Fuente.*

Los datos expresados en notación hexadecimal, no tienen de por sí ningún significado para el Spectrum ya que éste es incapaz de trabajar con números que no sean decimales o binarios. Previamente a su utilización, el Código Fuente deberá transformarse en números decimales para que puedan ser entendidos perfectamente por el Ordenador. Esto es lo que llamamos *Código Objeto.*

Esta operación de transformar el Código Fuente (Datos hexadecimales) en Código Objeto se llama «DUMPING» (Volcado en memoria) y la hace automáticamente nuestro programa mediante el comando «Dump».

Una vez tecleado el Programa Cargador, hay que hacer GOTO 9900, con lo que se grabará y verificará en cinta.

El programa se pondrá en funcionamiento automáticamente. Si por cualquier razon, intencionada o no, se detuviese durante su utilización, es imprescindible teclear, «GO TO menú», nunca RUN ni ningún tipo de CLEAR ya que estos dos comandos destruyen las variables y con ellas, el Código Fuente que hubiera almacenado hasta el momento.

Utilización

Una vez cargado desde la cinta, el programa se pondrá en marcha automáticamente, presentando en la línea inferior de la pantalla, un pequeño menú de opciones, a cada una de las cuales se accede pulsando la tecla que corresponde con su inicial.

INPUT. Este comando sirve para introducir nuevas líneas de Código Fuente. Al pulsarlo, el programa nos solicita un número de línea. Obligatoriamente, hemos de comenzar por la línea 1, a no ser que ya hayamos introducido alguna otra previamente.

Tras indicar el número de línea, nos pedirá los Datos correspondientes a la misma. Una vez tecleados (observese un trazo grueso negro que nos ayuda a controlar que el número de caracteres alfanuméricos introducidos sea 20 en todos los casos) y suponiendo que no haya habido ningún error hasta el momento, hay que introducir el Control, que está situado, en cada Línea, a la derecha del Dato. Por último, el programa nos solicita una nueva línea, lo que nos da a entender que todo el proceso anterior ha sido correcto.

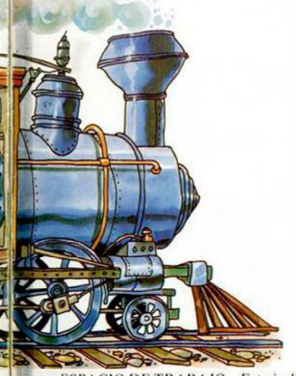
En el momento en que se nos solicita nueva Línea o cuando se nos pide el Dato podemos pasar, si lo deseamos, al menú principal pulsando simplemente «ENTER».

TEST. Tiene el doble cometido de listar por pantalla las líneas de Datos que hallamos metido hasta el momento, y de averiguar si una determinada línea ha sido admitida como correcta, ya que si no ha sido aceptada, tampoco aparecerá en el listado.

DUMP. Este comando vuelca el contenido de la variable a$ en memoria, a partir de la Dirección que se especifique. Equivale a convertir el Código Fuente en Código Objeto. Esta operación es obligatoria antes de hacer funcionar una rutina o programa en código máquina. En la mayoría de los casos, con la rutina se indicará también la dirección de memoria donde deber ser volcada y su longitud expresada en bytes. Cuando no se especifique dirección alguna es que la rutina o programa pueden funcionar en cualquier parte de la memoria.

En algún caso puede ocurrir que al intentar volcar el Código Fuente en la memoria, nos aparezca el mensaje

```
   2 REM
   3 REM CARGADOR CM MICROHOBBY
   4 REM
   5 CLEAR 65535: LET menu=6000
  10 FOR n=23296 TO 23312
  12 READ c: POKE n,c: NEXT n
  15 DATA 42,75,92,126,254,193,4
0,6,205,184,25,235,24,245,54,65,
201
  70 LET a$="": POKE 23658,8
 100 LET a=10: LET b=11: LET c=1
2: LET d=13: LET e=14: LET f=15
 200 LET li=1: GO TO 6000
1000 REM BUCLE PRINCIPAL
1001 INPUT "LINEA", LINE l$: IF
l$="" THEN GO TO 6000
1002 FOR n=1 TO LEN l$
1003 IF l$(n)<"0" OR l$(n)>"9" T
HEN GO TO 1001
1004 NEXT n: LET line=VAL l$
1005 IF line<>li THEN POKE 23689
,PEEK 23689-1: GO SUB 5000: GO T
O 1000
1007 INPUT "
              DATOS    "; LINE d$
1008 IF d$="" THEN GO TO 6000
1009 LET cx=24-PEEK 23689: PRINT
 AT cx,0;d$;AT cx,21;CHR$ 138;"L
INEA ";li
1010 IF LEN d$<>20 THEN GO SUB 5
000: GO TO 1000
1020 FOR n=1 TO 20
1110 LET w$=d$(n)
1150 IF w$>CHR$ 47 AND w$<CHR$ 5
8 OR w$>CHR$ 64 AND w$<CHR$ 71 T
HEN GO TO 1170
1160 PRINT AT cx,n-1; FLASH 1; O
VER 1;" ": GO SUB 5000: GO TO 10
00
1200 NEXT n: LET ch=0
1210 FOR n=1 TO 20 STEP 2
1215 LET he=VAL d$(n)*16+VAL d$(
n+1): LET ch=ch+he: NEXT n
1250 LET ct=0: INPUT "CONTROL ",
ct
1260 IF ct<>ch THEN GO SUB 5000:
 GO TO 1000
1300 LET a$=a$+d$
2000 LET li=li+1: GO TO 1000
5000 BEEP .2,0: OUT 254,2: POKE
23689,PEEK 23689+1: RETURN
6000 REM MENU PRINCIPAL
6005 PRINT #0; INK 7; PAPER 1;"
   INPUT LOAD SAVE DUMP TEST    "
```



«ESPACIO DE TRABAJO». Esto indica que estamos intentando volcar en una zona que el ordenador está usando para sus propios cálculos. Volcar ahí significa la destrucción del programa y todo nuestro trabajo. En tales casos, no queda más remedio que elegir otra dirección de volcado más apropiada.

Durante el tiempo que dura la operación de volcado (depende de la longitud del Código Fuente), se nos muestra en pantalla la dirección inicial y las que restan en ese momento.

SAVE. Este comando nos permite salvar en cinta el Código Fuente (muy importante cuando dejemos el trabajo de tecleo a medias) o el Código Objeto (también llamado Código Máquina) para su posterior utilización. Al pulsar SAVE nos aparecerá un segundo menú de tres opciones: Salvar Código Fuente (F), Salvar Código Objeto (O), indicando Dirección y número de bytes, o volver al menú principal (R). En los dos primeros casos hay que especificar el nombre con el que queremos salvar el Código.

Todas las operaciones de SAVE pueden, opcionalmente, verificarse.

Es muy importante recordar que nunca podrá utilizarse no salvarse el Código Objeto si antes no se ha procedido a su volcado en memoria mediante el comando DUMP.

LOAS. Cuando el número de Datos a teclear sea grande, es normal tener que realizar el trabajo en varias veces. Para ello, puede salvarse en cinta la parte que tengamos (Código Objeto) y luego recuperar mediante la opción LOAD. No es necesario indicar nombre en este caso si no se recuerda. Al cargarse correctamente el Código Fuente, el ordenador nos indicará automáticamente cual fue la última línea que habiamos tecleado y cual es la primera que hemos de introducir ahora.

Los errores

El Cargador de Código Máquina está especialmente estructurado para tratar de prevenir todos los errores típicos de la introducción de datos y que, en el caso concreto de los programas en Código Máquina tienen, por lo general, consecuencias desastrosas, dando al traste con horas e incluso dias de trabajo.

El programa que presentamos chequea las siguientes posibilidades de error:

Que el número de línea no sea correlativo, en cuyo caso se trataría, sin duda, de un error de omisión de línea. Es decir, después de la línea 2, tiene que venir la 3, y no otra.

Que la longitud de la cadena de Datos sea 20. Si es mayor o menor es que sobran o faltan dígitos.

Que las cifras introducidas dentro de una línea de Datos no esten comprendidas dentro del rango de los caracteres utilizados en la notación hexadecimal. Es decir, entre 0 y F. Cualquier anomalía en este sentido será inmediatamente indicada con el parpadeo de la cifra errónea.

Que el control no coincida con la suma de los valores de los Datos en decimal. (Cada dos Datos forman un número hexadecimal).

En todos estos casos, el ordenador nos advierte del error con una señal acústica, a la vez que el borde de la pantalla se vuelve rojo. En situación normal (mientras no se produce ningún error) el borde deberá permanecer siempre blanco.

También hay que tener en cuenta que cualquier error anula la validez de la línea en curso, por lo que habra que repetirla de nuevo correctamente. Para saber las líneas aceptadas en todo momento pulsar Test. A partir de la última, hay que continuar introduciendo nuevas líneas.

Un ejemplo práctico

Como ejemplo de utilización puede valer esta pequeña rutina de Código Máquina que sirve para borrar la pantalla lateralmente. Teclea el Código Fuente. Haz DUMP en la dirección 40000. Salva el Código Objeto desde la dirección 40000, 45 bytes. Haz «BREAK» con «CAPS SHIFT» y «SPACE». Para probar que tanto la Rutina como el Cargador funcionan correctamente, teclea RANDOMIZE USR 40000. La pantalla se verá invadida lateralmente por una cortina azul. Puedes usar esta Rutina en tus propios programas haciendo previamente LOAD " " CODE y llamándola luego desde Basic con RANCOMIZE USR 40000.



```
3100 LET i$=INKEY$: IF i$="" THE
N GO TO 6100
6200 IF i$="I" THEN GO TO 1000
6210 IF i$="S" THEN GO TO 7000
6220 IF i$="L" THEN GO TO 8000
6225 IF i$="T" THEN GO TO 7500
6230 IF i$="D" THEN GO TO 9000
6250 GO TO 6100
7000 REM SAVE
7001 PRINT #0; PAPER 3; INK 7;"
 FUENTE(F) OBJETO(O) RETURN(R) "
7002 PAUSE 0: IF INKEY$<>"F" AND
 INKEY$<>"O" AND INKEY$<>"R" THE
N GO TO 7002
7003 IF INKEY$="O" THEN GO TO 72
50
7004 IF INKEY$="R" THEN CLS : GO
 TO 6000
7005 REM SAVE DATA
7006 IF a$="" THEN GO SUB 9500:
GO TO 6000
7008 RANDOMIZE li
7010 LET a$=CHR$ PEEK 23670+CHR$
 PEEK 23671+a$
7015 INPUT "NOMBRE (Save)", LINE
 n$: IF n$="" OR LEN n$>10 THEN
GO TO 7015
7020 SAVE n$ DATA a$()
7025 PRINT #0; PAPER 6;"     DES
EA VERIFICAR (S/N)        ": PAUSE
 0: IF INKEY$="S" THEN PRINT #0;
 INK 7; PAPER 2;" REBOBINE LA CI
NTA Y PULSE PLAY ": VERIFY n$ DA
TA a$(): CLS : PRINT "CODIGO FUE
NTE: ";n$: PAUSE 200
7030 LET a$=a$(3 TO ): CLS
7040 GO TO 6000
7250 REM SAVE CODE
7255 INPUT PAPER 3; INK 7;"DIREC
CION ";di, PAPER 3; INK 7;"N.BYT
ES ";nb
7260 INPUT "NOMBRE (Save)", LINE
 n$: IF n$="" OR LEN n$>10 THEN
GO TO 7260
7270 SAVE n$CODE di,nb
7275 PRINT #0; PAPER 6;"     DES
EA VERIFICAR (S/N)        ": PAUSE
 0: IF INKEY$="S" THEN PRINT #0;
 INK 7; PAPER 2;" REBOBINE LA CI
NTA Y PULSE PLAY ": VERIFY n$COD
E di,nb: CLS : PRINT "CODIGO OBJ
ETO: ";n$'"Inicio: ";di,"Longitu
d: ";nb: PAUSE 200
```



```
7280 CLS
7300 GO TO 6000
7500 REM TEST
7503 IF a$="" THEN GO SUB 9500:
GO TO 6000
7505 CLS : FOR m=1 TO (LEN a$) S
TEP 20
7510 PRINT a$(m TO m+19);" ";CHR
$ 138;"LINEA ";INT (m/20)+1
7520 NEXT m: GO TO 6000
8000 REM LOAD
8010 INPUT "NOMBRE (Load)", LINE
 n$
8020 LOAD n$ DATA a$()
8025 RANDOMIZE USR 23296
8030 LET li=CODE a$(1)+256*CODE
a$(2): LET a$=a$(3 TO )
8035 CLS : PRINT AT 10,5;"Ultima
 linea: ";li-1;AT 11,5;"Comenzar
 por: ";li
8040 GO TO 6000
9000 REM DUMPING
9003 IF a$="" THEN GO SUB 9500:
GO TO 6000
9005 INPUT "DIRECCION ",di: CLS
9006 IF di<PEEK 23653+256*PEEK 2
3654 OR (di+LEN a$/2)>65300 THEN
 PRINT FLASH 1;AT 5,6;"ESPACIO D
E TRABAJO": FOR n=1 TO 200: NEXT
 n: CLS : GO TO 6000
9007 PRINT AT 9,7; FLASH 1;"VOLC
ANDO EN MEMORIA": PRINT AT 7,5;"
Direccion Inicial: ";di
9008 PRINT AT 11,4;"RESTAN ";AT
11,17;"DIRECCIONES"
9010 FOR n=1 TO (LEN a$) STEP 2
9015 POKE di,VAL a$(n)*16+VAL a$
(n+1): LET di=di+1
9018 PRINT AT 11,12;INT (LEN a$/
2-n/2);" "
9020 NEXT n: CLS : PRINT AT 10,8
; FLASH 1;"VOLCADO COMPLETO": FO
R n=1 TO 100: NEXT n: CLS : GO T
O 6000
9500 REM SUBRUTINA MEMORIA VACIA
9501 CLS : PRINT #0; FLASH 1;" N
O EXISTE NINGUN CODIGO FUENTE ":
 PAUSE 300: CLS : RETURN
9900 CLEAR : SAVE "CARGADOR" LIN
E 1: PRINT #0;"REBOBINE LA CINTA
 PARA VERIFICAR": VERIFY "CARGAD
OR": RUN
```

```
1  0E202100580618112000 246
2  E536121910FBC521D000 1031
3  112000CDB503C1E1E506 1091
4  1811200036091910FBE1 653
5  23002009C90000000000 498
```

COPIADOR DE CARACTERES

La presente rutina se encarga de una función que pocas utilidades de las de «dibujar» poseen. Con ella podremos, operando con una pantalla, coger de cualquier posición un carácter, guardarlo en un buffer, y hacer todas las copias que deseemos del mismo en cualquier parte de la pantalla.

Nada más llamar tendremos un cursor parpadeante en la esquina superior izquierda de la pantalla, éste lo podremos mover con las teclas del cursor, y cuando esté encima del carácter que deseemos copiar pulsaremos el «9», en este momento el borde se pondrá de color verde, indicando esto que la última operación que hemos hecho es la de coger un carácter. Luego bastará desplazarse a la posición de carácter donde deseemos hacer la copia y pulsar el «0» el borde se pondrá de color rojo indicando que la última operación que hemos hecho es la de pintar.

Para volver a Basic sólo tendremos que pulsar el «Space».

Para aquellos que no dispongan de un ensamblador código máquina o no posean conocimientos de éste, el listado lo hemos reproducido en el modelo del cargador universal que aparece en las páginas anteriores.

LISTADO 1

```
10 ; PROGRAMA COPIADOR
20 ; DE CARACTERES
30 ; POR J.M.LAZO
40
50        ORG  50000
60        ENT  $
70        LD   A,0
80        LD   (COORDY),A
90        LD   (COORDX),A
100       LD   (ATRIB),A
110       LD   B,8
120       LD   HL,BUFFER
130 LOOP1 LD   (HL),0
140       INC  HL
150       DJNZ LOOP1
160 TECLAS LD  A,247
170       IN   A,(#FE)
180       BIT  4,A
190       CALL Z,IZQUI
200       LD   A,239
210       IN   A,(#FE)
220       BIT  4,A
230       PUSH AF
240       CALL Z,ABAJO
250       POP  AF
260       BIT  3,A
270       PUSH AF
280       CALL Z,ARRIBA
290       POP  AF
300       BIT  2,A
310       PUSH AF
320       CALL Z,DERE
330       POP  AF
340       BIT  1,A
350       PUSH AF
360       CALL Z,PINTA
370       POP  AF
380       BIT  0,A
390       CALL Z,SACA
400       LD   A,127
410       IN   A,(#FE)
420       BIT  0,A
430       RET  Z
440       JP   CURSOR
450 IZQUI LD   A,(COORDX)
460       DEC  A
470       CP   255
480       JR   NZ,REES
490       INC  A
500 REES  LD   (COORDX),A
510       RET
520 ARRIBA LD  A,(COORDY)
530       DEC  A
540       CP   255
550       JR   NZ,REES1
560       INC  A
570 REES1 LD   (COORDY),A
580       RET
590 ABAJO LD   A,(COORDY)
600       INC  A
610       CP   24
620       JR   NZ,REES2
630       DEC  A
640 REES2 LD   (COORDY),A
650       RET
660 DERE  LD   A,(COORDX)
670       INC  A
680       CP   32
690       JR   NZ,REES3
700       DEC  A
710 REES3 LD   (COORDX),A
720       RET
730 COORDX DEFB 0
740 COORDY DEFB 0
750 ATRIB  DEFB 0
760 BUFFER DEFS 8
770 CURSOR CALL POSIC
780       LD   A,(HL)
790       LD   (ATRIB),A
800       LD   (HL),127
810       CALL PAUSA
820       LD   A,(ATRIB)
830       LD   (HL),A
840       CALL PAUSA
850       JP   TECLAS
860 POSIC LD   HL,22528
870       LD   A,(COORDY)
880       LD   E,A
890       LD   D,0
900       LD   B,32
910 LOOP2 ADD  HL,DE
920       DJNZ LOOP2
930       LD   A,(COORDX)
940       LD   E,A
950       LD   D,0
960       ADD  HL,DE
970       RET
980 PINTA LD   A,(COORDX)
990       LD   C,A
1000      LD   A,(COORDY)
1010      LD   B,A
1020      CALL COORD
1030      JR   SIGUE
1040 COORD LD  H,#40
1050      LD   A,B
1060      AND  24
1070      ADD  A,H
1080      LD   H,A
1090      LD   A,B
1100      AND  7
1110      RRCA
1120      RRCA
1130      RRCA
1140      ADD  A,C
1150      LD   L,A
1160      RET
1170 SIGUE LD  B,8
1180      LD   DE,BUFFER
1190 LOOP3 LD  A,(HL)
1200      INC  H
1210      LD   (DE),A
1220      INC  DE
1230      DJNZ LOOP3
1240      LD   A,4
1250      OUT  (#FE),A
1260      RET
1270 SACA LD   A,(COORDX)
1280      LD   C,A
1290      LD   A,(COORDY)
1300      LD   B,A
1310      CALL COORD
1320      LD   B,8
1330      LD   DE,BUFFER
1340 LOOP4 LD  A,(DE)
1350      INC  DE
1360      LD   (HL),A
1370      INC  H
1380      DJNZ LOOP4
1390      LD   A,2
1400      OUT  (#FE),A
1410      RET
1420 PAUSA LD  BC,5000
1430 LOOP5 DEC BC
1440      LD   A,B
1450      OR   C
1460      JR   NZ,LOOP5
1470      RET
1480 ZINAL
```

LISTADO 2

```
1   3E0032D2C332D1C332D3 1232
2   C3060821D4C336002310 754
3   FB3EF7DBFECB67CC9DC3 1895
4   3EEFDBFECB67F5CCB7C3 1907
5   F1CB5FF5CCAAC3F1CB57 1884
6   F5CCC4C3F1CB4FF5CC08 1820
7   C4F1CB47CC35C43E7FDB 1572
8   FECB47C8C3DCC33AD1C3 1800
9   3DFEFF20013C32D1C3C9 1318
10  3AD2C33DFEFF20013C32 1176
11  D2C3C93AD2C33CFE1820 1439
12  013D32D2C3C93AD1C33C 1240
13  FE2020013D32D1C3C900 1035
14  00000000000000000000 0
15  CDF2C37E32D3C3367FCD 1610
16  50C43AD3C377CD50C4C3 1535
17  65C32100583AD2C35F16 997
18  0006201910FD3AD1C35F 889
19  160019C93AD1C34F3AD2 1057
20  C347CD15C41810264078 950
21  E618846778E6070F0F0F 891
22  816FC9060811D4C37E24 1041
23  121310FA3E04D3FEC93A 1093
24  D1C34F3AD2C347CD15C4 1439
25  060811D4C31A13772410 654
26  FA3E02D3FEC901881308 1147
27  78B120FBC90000000000 781
```